NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

USING COMMERCIAL OFF THE SHELF (COTS)
DIGITAL SIGNAL PROCESSORS (DSP) FOR RELIABLE
SPACE BASED DIGITAL SIGNAL PROCESSING

by

Matthew J. Wukitch

March 2001

Thesis Co-Advisor: Herschel H. Loomis
Alan A. Ross

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave Blank)

2. REPORT DATE
March 2001

3. REPORT TYPE AND DATES COVERED
Master's Thesis

4. TITLE AND SUBTITLE
Using Commercial off the shelf (COTS) Digital Signal Processors (DSP) for Reliable Space Based Digital Signal Processing

5. FUNDING NUMBERS

6. AUTHOR(S)
Wukitch, Matthew J.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

3. ABSTRACT *(Maximum 200 words)*

A radiation tolerant testbed was designed using a Commercial-Off-the-Self Digital Signal Processor and presented to prove the concept of Triple Modular Redundant (TMR) processors in order to make a COTS DSP radiation tolerant design. The system was designed to handle the effects of radiation associated with Single Event Upset only.

Two of the industry's leading programmable 32-bit floating-point digital signal processors were reviewed for this thesis, Analog Devices ADSP-21060 and the Texas Instruments TMS320C6701. The '6701 was the best processor for this design based upon size, power, speed, and tolerance to single event latchup, signal event burnout, and total ionization dose. A review of the processor's performance and characteristics is provided to ensure the proper operation of '6701 in a TMR design.

The system employs a bit by bit voter that compares the three processors' results and outputs the majority of the bits. All data, address, and control signals are monitored to determine that the system is operating properly. This system significantly differs from previous TMR designs, because only address errors cause immediate interrupts. Data errors cause processor interrupts only when the errors accumulate to a critical level. An external host processor controls the processors' shared memory space.

14. SUBJECT TERMS
Fault-tolerant Computing, Digital Signal Processors, Texas Instruments TMS320C6701, Commercial-off-the-Shelf Technology, Radiation, Triple Modular Redundant, Analog Devices ADSP-21060

15. NUMBER OF PAGES
120

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT
Unclassified

18. SECURITY CLASSIFICATION OF THIS PAGE
Unclassified

19. SECURITY CLASSIFICATION OF ABSTRACT
Unclassified

20. LIMITATION OF ABSTRACT
UL

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

USING COMMERCIAL OFF THE SHELF (COTS) DIGITAL SIGNAL PROCESSORS (DSP) FOR RELIABLE SPACE BASED DIGITAL SIGNAL PROCESSING

Matthew J. Wukitch
Lieutenant, United States Navy
B.S., U.S. Naval Academy, 1994

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL

March 2001

Author: ______________________
Matthew J. Wukitch

Approved by: Herschel H Loomis, Jr
Herschel H. Loomis, Thesis Co-Advisor

Alan A Ross
Alan A. Ross, Thesis Co-Advisor

____
Jeffrey B. Knorr, Chairman
Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

A radiation tolerant testbed was designed using a Commercial-Off-the-Self Digital Signal Processor and presented to prove the concept of Triple Modular Redundant (TMR) processors in order to make a COTS DSP radiation tolerant design. The system was designed to handle the effects of radiation associated with Single Event Upset only.

Two of the industry's leading programmable 32-bit floating-point digital signal processors were reviewed for this thesis, Analog Devices ADSP-21060 and the Texas Instruments TMS320C6701. The '6701 was the best processor for this design based upon size, power, speed, and tolerance to single event latchup, signal event burnout, and total ionization dose. A review of the processor's performance and characteristics is provided to ensure the proper operation of '6701 in a TMR design.

The system employs a bit by bit voter that compares the three processors' results and outputs the majority of the bits. All data, address, and control signals are monitored to determine that the system is operating properly. This system significantly differs from previous TMR designs, because only address errors cause immediate interrupts. Data errors cause processor interrupts only when the errors accumulate to a critical level. An external host processor controls the processors' shared memory space.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

First I would like to thank my mom and dad for always being there when I need them and Greg for always helping me out and taking trips to Canada. I would like to thank the following people for helping me to complete my thesis. Prof. Loomis and Prof. Ross for providing guidance. Jeff Weekley for editing and correcting my horrible grammar. I would like thank the following groups, Reel Big Fish, Less than Jake, Goldfinger, Catch 22, 311, Everclear, Third Eye Blind, Greenday, Blink 182 and all the other great bands that kept me sane will working on this project.

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

Radiation tolerant digital signal processors have not kept pace technologically with other digital signal processors that do not need to operate in the rigorous environment of space. The traditional approach of radiation hardening has not allowed for the inexpensive and quick development of designs suitable for space. This thesis offers one solution to the difficulties of the space deployment of digital signal processors, while providing for less expensive components and more flexibility in the end product application.

A radiation tolerant testbed was designed using a Commercial-Off-the-Self Digital Signal Processor and presented to prove the concept of Triple Modular Redundant (TMR) processors in order to make a COTS DSP radiation tolerant design. The system was designed to handle the effects of radiation associated with Single Event Upset only.

Two of the industry's leading programmable 32-bit floating-point digital signal processors were reviewed for this thesis, Analog Devices ADSP-21060 and the Texas Instruments TMS320C6701. The '6701 was the best processor for this design based upon size, power, speed, and tolerance to single event latchup, signal event burnout, and total ionization dose. A review of the processor's performance and characteristics is provided to ensure the proper operation of '6701 in a TMR design.

The system employs a bit by bit voter that compares the three processors' results and outputs the majority of the bits. All data, address, and control signals are monitored to determine that the system is operating properly. This system significantly differs from previous TMR designs, because only address errors cause immediate interrupts. Data errors

cause processor interrupts only when the errors accumulate to a critical level. An external host processor controls the processors' shared memory space.

I. INTRODUCTION

A. BACKGROUND

Digital signal processing is an area of study that has advanced rapidly over the past 30 years. The rapid development of very-large-scale integration (VLSI) of electronic circuits has spurred the development of more powerful, smaller, faster, and cheaper digital signal processors. Special purpose signal processors were introduced in the 1980s and advanced rapidly during the 1990s. The first processors were simple 8-bit fixed point processors that have advanced to today's 32-bit fixed or floating-point processors that can perform up to 1.1 billion instructions per second. The advantages of using digital signal processing over analog techniques include increased stability, reduced noise susceptibility, re-programmability, and high immunity to factors such as component aging, loss of precision, and temperature variations. These inexpensive and relatively fast processors have made it possible to build highly sophisticated digital systems capable of performing complex digital signal processing functions and tasks. Hence many of the signal processing tasks that were performed by analog circuits can now be done cheaper, more reliably and with decreased power requirements with a digital signal processor implementation. Therefore, it was natural to adapt these processors for space-based applications that range from signal processing, image processing, attitude control, and power management [Ref. 1].

B. SPACE RADIATION ENVIROMENT

The space environment requires a designer to take special precautions that are not required inside the earth's atmosphere. This environment poses a risk to all earth-orbiting satellites and missions to other planets in the form of electromagnetic radiation from the sun: not only visible light, but the entire range from radio to gamma rays. In addition to solar radiation, a satellite design has to deal with ionization particles of the sun and distant stars. Some of this is trapped within the earth's magnetic field forming the intense radiation of the Van Allen Belts [Ref. 2].

Radiation is the process of emitting radiant energy in the from of waves or particles. Most of the radiation in space near to Earth comes from the sun, as fusion in the sun shoots particles and radiates energy in the form of waves. The particles emitted by the sun become deflected or trapped by the Earth's magnetic field. The harmful high-energy waves are absorbed by particles in the upper atmosphere [Ref. 2].

Particles can come in a variety of forms classified in two groups, heavy and light ions. The light ion will be either a proton or a beta particle (electron). The proton is the simplest positive ion and is a fundamental particle with a small mass. The beta particle is the simplest negatively charged ion and also has a small mass. Stripping the two electrons from a helium atom creates the alpha particle. The classifications of ions as heavy or light is dependent on the atomic number of the element. All ions with two or more protons/neutrons are classified as heavy ions [Ref. 3].

Unlike particles emitted by the sun and other celestial objects the waves or photons emitted by them have neither mass nor charge. Gamma Rays are an example of photon radiation.

The sun dominates the sources of radiation in space near earth. Solar activity on the sun varies over an 11-year cycle, producing a variable average of solar particles. Though solar activity is predictable on a macro scale, the sun still produces a wide variation in radiation intensity on a day-to-day basis [Ref. 3].

A second type of particle radiation is the Galactic Cosmic Ray (GCR), which consists of are particles that reach earth from sources outside the solar system. Cosmic rays are heavy ions produced from cosmic events such as exploding stars.

The largest contributor to a spacecraft's total dose is particles trapped by the earth's magnetic field. These particles come mainly from the sun but also from other celestial bodies. The particles create the region in space known as the Van Allen Belts. The belts are a fixed hazard to spacecraft and are distributed non-uniformly within the magnetosphere.

From these sources of radiation two factors, Total Ionization Dosage (TID) and Single Event Effects (SEE) are calculated to determine the survivability of spacecraft and their electronics. The Total Ionization Dosage (TID) is the total amount of radiation that can be absorbed during a spacecraft's lifetime before failure. TID creates bulk-oxide and interface-trap charge that reduces transistor gain and shifts the operating properties (e.g., threshold voltage) of semiconductor devices. TID accumulation will cause a device to fail in the following ways: the transistor threshold voltage shifts far enough to cause circuit malfunction; the device fails to operate at the required frequency; or electrical isolation between devices is lost. All electronic devices experience the effects of TID, and these effects cannot be mitigated after the design and manufacturing process. Therefore, we will focus on the second type of radiation effects, the single event effects.

Single Event Effects (SEE) consists of three different phenomena. They are single event upset (SEU), single event latchup (SEL) and the single event burnout (SEB). SEE are random events that occur when spacecraft are exposed to radiation. As the spacecraft orbit the earth, it is continuously exposed to radiation, some of which passes through their external shell and affect the spacecraft subsystems. When radiation interacts with microelectronic devices, it can cause behaviors that are unpredictable and potentially damaging. They are discussed in detail in the following sections.

C. SINGLE EVENT EFFECTS

SEEs occur when high-energy particles pass through microelectronic device and deposit enough energy to cause a transistor to change state. In most cases this is a transient effect occurring only long enough for the charge to be absorbed by the system. The transistor's state change can lead to latchup in parasitic transistors, high current state in a power transistor, or can be latched into a storage element. These three main types of SEE in complimentary Metal Oxide Semiconductors (CMOS) are discussed in the following sections [Ref. 3].

1. Single Event Upset

An SEU is an unpredicted change of state or "bit flip" caused by a high-energy particle passing through a device. In a microprocessor a bit flip could cause a processor to branch to an unexpected location in memory thus causing a program to crash. In microprocessors, SEUs are typically grouped into two types: program run errors and data errors. Program run errors are errors that occur in control logic, program counter, or any other register that determines the state of processor. Data errors are typically confined to data registers and cache. These two types of error are not necessarily exclusive. A data error could occur in a register that is later used as a program address. When the microprocessor reads the address held in that register it is going to run the wrong code located at that address or even an invalid address [Ref. 3].

2. Single Event Latchup (SEL)

Integrated circuits are made by combining adjacent p-type and n-type regions into transistors. By the nature of the process, parasitic transistors are formed along alternate paths through the circuit. Normally, these parasitic transistors are biased off by the circuit design. Latchup occurs when the charge, such as that produced by a particle, activates one of these parasitic transistors, which forms into a circuit with large positive feedback. This creates a short circuit across the device, with two possible outcomes. The first is the current drawn through the parasitic transistor generates more heat than the device can dissipate and burns out. If the device is able to dissipate the heat, the large amount of current drawn through the parasitic transistor prevents the circuit from functioning correctly, which is a non-destructive SEL. The normal symptom of a non-destructive SEL is a hung system, which requires power to be cycled before proper operations can be restored [Ref. 3].

3. Single Event Burnout (SEB)

SEB is another condition that can cause device destruction. It is caused by a single ion that induces a high current state in a MOSFET destroying the circuit [Ref. 3].

D. PURPOSE

One approach to building radiation tolerant electronics is to design and build them on an electrical insulator. The process of building on an electrically insulator is often called radiation hardening (Radhard). Another approach to fault tolerant computing is to design a fault tolerant architecture. The goal of this research is the testing and implementation of a fault tolerant computer system using a COTS programmable digital signal processor that is capable of operating in the presence of radiation induced SEUs.

This design does not take into account total dose radiation, which is a factor that usually limits the operational lifetime of spacecraft electronics. This factor is determined by the electrical properties of solid-state components exposed to radiation over a period of time. Ultimately the long exposure to radiation leads to changes in the component parameters outside of design specifications and causes the circuit to cease proper functioning. This factor must be addressed by spacecraft shielding, component selection and survivability, and not by system architecture. Other factors that must be addressed are SEL and SEB. These two phenomena can be limited by external power monitoring devices. Implementation of power monitoring devices will be left to follow-on studies.

Successful completion of this project will lead to numerous benefits for the space community. First, the adage of faster better cheaper can be utilized in the development of spacecraft. The spacecraft engineer will have a broader choice of devices and software to choose from at a reduced cost. The spacecraft design will no longer be restricted to the use of radhard components.

Second, the fault tolerant system can be utilized as a testbed to analyze software fault tolerant programs. The fault tolerance hardware is able to detect the SEU and log the time and kind of an upset. The software can then be observed in the manner in which it handles the error. This testbed will allow the testing of software in a simulated space environment prior to use in orbit.

Last, the system can be utilized as a hybrid fault tolerant computer system. In this configuration, the processor is additionally monitored for SEU. Upon detecting an upset, the processor is restored to the state prior to the upset. The processor then continues execution from the point prior to the upset with little downtime and no loss of data. This is dramatically different from current operations where a processor is reset when an error occurs, resulting in downtime, loss of data and/or spacecraft availability. As shown, this is a major advance in the handling of spacecraft system failures.

E. RADHARD VERSUS COTS

The radiation effects discussed in the previous sections, with the exception of SEUs, are destructive in nature. The main way of reducing their effects is by utilizing radiation hardened (radhard) devices or by providing shielding. A radhard device is one that is specifically designed to be able to withstand higher amounts of radiation than standard commercial parts.

Determining the suitability of commercial-off-the-shelf (COTS) signal processor for space applications is a subject of ongoing research. To adapt COTS devices for space a limited amount of radiation hardening is necessary to resist the effects of TID, SEL, and SEB. There are multiple reasons for utilizing a COTS device in the harsh environment of space. This section will present a few of the rationale leading to the use of COTS.

1. Cost of Development of Radhard Devices

The small percentage of the overall market that requires radhard components puts severe economic constraints on the companies that produce these devices. The number of companies developing and marketing radhard devices is rapidly on the decline and the remaining companies are not developing new chip designs. For these reasons, the development of radhard devices is lagging behind state of the art technology by two or more generations. As an example, a spacecraft launched into space at present would have at best the equivalent of a Analog Devices ADSP-21020 25 MHz DSP that can execute 50 MFLOPS sustained performance compared to the current generation Texas Instrument (TI™) TMS320C6701 167 MHz processor that can execute sustained performance 668 MFLOPS. This entire order of magnitude difference in processor capability makes the COTS processor especially appealing [Ref. 8 and 9].

a. High development Cost of Radhard Devices

Low demand and little profit exist in the production of radhard devices, which has led to many manufacturers abandoning their production of radhard devices in favor of the more lucrative, higher volume consumer electronics. The limited availability of these devices then leads to an inflation of the cost. Part cost directly impacts the cost of the product. In a time of shrinking budgets, the spacecraft engineer is looking for a cheaper suitable product. The best alternative is the development of hardware and software fault tolerant designs with non-radhard COTS [Ref. 3].

To produce the radhard ADSP-21020 (RH-21020) required 22 months from the beginning of the process to having a working prototype for delivery. This was a successful transfer of a commercial design to a radhard process. There was a tremendous amount of money and resources that were dedicated by the Air Force Research Laboratory (AFRL), Analog Devices and Lockheed Martin to jointly transfer the technology from standard silicon design to a radhard design. Though this was successful, it was very risky and had it not succeeded, all of those resources would have been wasted. Even with the backing of the AFRL the program for the technology transfer ran short of funding and a complete debugging of test programs and test patterns could not be completed [Ref. 8].

b. Cost for Software

Even the use of a commercial design for a radhard process does not guarantee that the software will be fully compatible. Advertised as a fully compatible family of processors, the earlier generations of ADSP-21xxx are actually of an older design, which have fewer functions without all the capabilities of the new processors in the normal 21xxx family. Therefore, modifications to the software will be required. Furthermore, the older design will require extensive testing, all of which adds to the cost of choosing a radhard design.

2. Benefits of Radhard

With the use of a radhard device, a known error rate can be calculated and compared with COTS. One error every ten years is achievable in radiation tolerant designs. This compares to a COTS device that can have an error rate of anywhere between several an hour to several a day or more [Ref. 9]. Radhard devices are also designed for the rigorous space environment of wide temperature swings and high vibrations. COTS devices may have a much narrow window of tolerances.

3. Cost of Using COTS

Since COTS devices are susceptible to radiation effects, a device will first need to be tested for its tolerance to destructive effects of radiation, such as TID, SEL and SEB. Once a suitable commercial product is found, its implementation will still require protection from SEU. This can be achieved either through software redundancy, hardware redundancy or a combination of the two. It is notable that a software redundancy is not always a useful solution for microprocessors, because each instruction would have to be executed at least twice and then results compared. This effectively cuts the processor speed by a minimum of one-half. Hardware redundancy also has its drawbacks. It requires an increase in power, circuit size and weight.

4. Benefits of COTS

The most apparent benefits of using COTS are lower cost of components, little or no need for special software, no restrictions on availability, faster processor speeds, and use of current design ICs. Because of the use of COTS we have eliminated the costly and time consuming design phase of transferring a technology to a radhard design and replaced it with a cheaper and faster screening process of radiation testing of commercial devices. By using current generation processors, the COTS processor has better availability and fastest processing speeds than radhard processors. There will also be extensive technical support by the manufacture that is not always available in older radhard processor designs.

F. RELATED WORK

The investigation into use of COTS is an on going research project at the Naval Postgraduate School. There are many ways to mitigate the harmful affects of a SEU. Triple Modular Redundant (TMR) systems are an accepted means of dealing with the affects of SEU; a detailed discussion follows in this thesis. Prior work includes the following theses. LT John Payne, USN, this work included the initial proposal for a TMR system for a microprocessor, based upon the IDT™ R3081 [Ref. 2]. Capt. Dave Summer's, USMC, thesis continued upon John's work by implementing the design into programmable logic devices and completing the design [Ref. 4]. Capt. Summer's work finalized the layout of the system architecture and it was built for testing. Capt. Susan Groening, USMC and LT Kimberly Whitehouse, USN designed the operating system and interrupt handling routine necessary to operate a TMR R3081 [Ref. 5]. LT Damen Hofhienz, USN, followed the previous work by verifying the layout and tested the printed TMR circuit board [Ref. 3]. LT Hofhienz' work also included recommendations for a space-flight-ready design as a follow on. LTjg Huseyin Ekin, Turkish Navy, is conducting research into adapting a 64-bit microprocessor vice the 32-bit microprocessor that was used in the previous TMR design. This thesis adapts the TMR design to a COTS 32-bit programmable digital signal processor.

G. THESIS ORGANIZATION

The organization of this thesis follows the design approach used in developing the system. Chapter I has been a brief introduction of the environment in which the system will be operating. Chapter II is a discussion on processor selection and justification. Chapter III considers effect of SEU and architectural solutions to corruption of the processor's operation. Chapter IV will present the triple modular redundant testbed design. Chapter V presents the conclusions developed during this research and discusses topics for follow-on work.

II. PROCESSOR SELECTION

A. CHARACTERISTICS

The logical place to start to design a general-purpose digital signal processing system is to set some system parameters. Since this DSP system is to be used as a design test-bed, it is unknown how it ultimately will be implemented. Therefore, the design includes as many possible configurations to allow a following designer greater flexibility. As a rule for this project as many functions of the digital signal processor as possible will be implemented. Some other basic requirements are that it must have floating point capabilities and be programmable.

B. GENERAL PURPOSE DIGITAL SIGNAL PROCESSOR VS GENERAL PURPOSE MICROPROCESSOR

So, why use a general purpose DSP over an already proven general-purpose microprocessor? The answer to this question requires a review of how DSP is performed. The core of DSP relies upon the use of the Fast Fourier Transform (FFT) and filtering, both of which have very repetitive simple mathematics structures. A FFT is shown in Figure 1 [Ref.1].

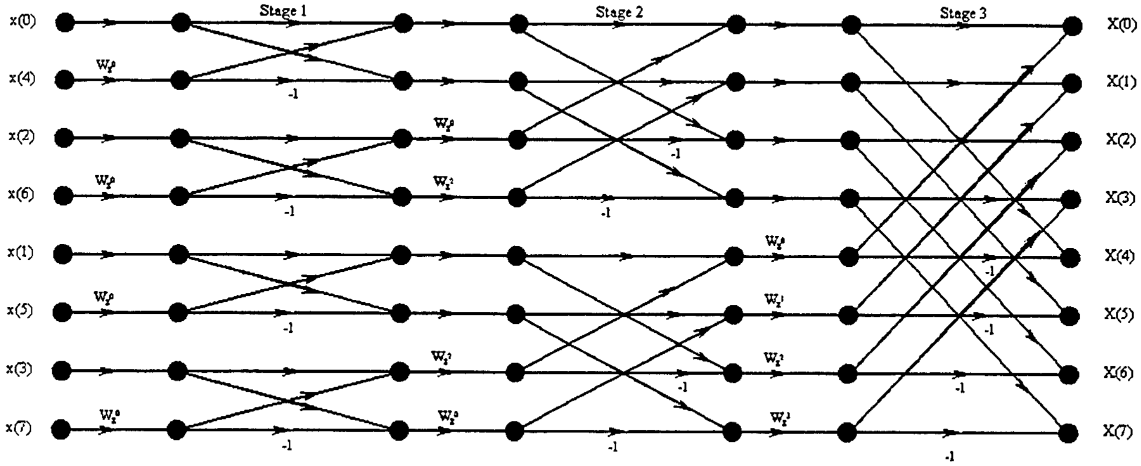

Figure 1. Eight Point Radix-2 FFT [From: Ref. 1].

An important aspect of the FFT structure is the repetitive nature of the algorithm. A closer look at the example of an eight point FFT in Figure 1 reveals that in order to complete stage one, 8 real multiplies and 12 real adds are required, resulting in 8 complex numbers. Stage two and beyond require 16 real multiplies and 40 real adds, resulting in 8 complex numbers. Table 1 demonstrates how much computational power is required to do the basic FFTs. To maximize throughput, it is best to keep as many of the objects being operated on either in registers or on-chip memory.

Real Multiplications				Real Additions		
n	Radix 2	Radix 4	Radix 8	Radix 2	Radix 4	Radix 8
16	24	20		152	148	
32	88			408		
64	264	208	204	1032	976	972
128	712			2504		
256	1800	1392		5896	5488	
512	4360		3204	13566		12420
1024	10248	7856		30726	28336	

Table 1. Real additions and Multiplications Required to Perform an N Point FFT [From: Ref. 1].

Since it is inevitable that results will need to be stored to memory, an efficient means of storage is necessary. With general-purpose processors, all addressing is either linear direct addressing or linear indirect addressing. In the FFT it is desirable to store data in a circular buffer, stepped through by a constant value. A general-purpose digital signal processor allows for circular addressing and removes the required overhead needed to achieve the same results as a general-purpose microprocessor would.

C. SIZE, PINOUT, POWER

In a design for space applications, the size, pinout and power consumption are important factors affecting a spacecraft's cost. A spacecraft can cost between 50,000-70,000 dollars per kilogram when placed into orbit. A higher power requirement for any given device demands increased amounts of solar cells and batteries. Therefore, this adds to the overall weight of the design and thus increases the costs. Likewise, the more pins required directly effects the size of the device and therefore more space will be required, increasing the weight of the spacecraft. These three factors are important to a spacecraft's design consideration and cannot be ignored [Ref. 2].

D. ON-CHIP MEMORY SIZE

It is very common in DSP that a short section of program code will be executed many times over a large group of data. In order to help decrease the processing time, it is important to be able to have single-cycle-access-times to memory in order to avoid halting the CPU to wait for data. On-chip memory often has multiple busses that allow for multiple access to internal memory. With multiple busses to memory, two different functional units can access two different memory locations in one clock-cycle. This is something that cannot be achieved with a single 32-bit external-memory-bus.

E. SPEED

The speed of a processor is not strictly determined by the clock rate; it is a combination of on-chip memory access time, CPU speed, ability to transfer large blocks of data and the speed with which the processor interacts with external devices and memory. With higher CPU clock speeds, a more robust set of applications can be performed. In analyzing speed performance for a DSP system, it is more important to focus on number of clock cycles (time) to execute a radix-2 FFT than it is to focus on how many floating-point operations per second (FLOPS).

F. PROCESSOR REVIEW

As part of this research, several digital signal processors were analyzed based on the discussion in the preceding section. Table 2 contains the data concerning the two final DSP that were considered in developing the testbed [Ref. 6].

Feature	TMS320C6701	ADSP-21060	Feature	TMS320C6701	ADSP-21060
IEEE Floating point support	Yes	Yes	**Accumulator Size**	80	80
Native 32 fixed point support	Yes	Yes	**64 bit product**	Yes	Yes
Internal SRAM size	1 Mbit	4 Mbit	**Register file to memory bandwidth**	128 bits/cycle	64 bits/cycle
Dual ported internal memory	No	Yes	**Software loops**	Required to be non interruptible for	Fully interruptible for
Number of serial ports	2	2	**Complexity of hand-optimized assembly**	Highly complex, must be written in non-single assignment form	Algebraic assembly language, easy to understand
Multiprocessing support	Supported in 2 ways, cluster and links	Supported in 2 ways, cluster and links	**Number of circular buffers supported**	8 (only two different lengths allowed, has to be power of two)	32 (different length for each buffer allowed)
DMA Channels	4	10	**Conditional execution**	Requires general purpose register	All instructions are conditional
Zero overhead DMA	No (there is cycle stealing from the	Yes	**FIR filter code size**	100 instructions (assumed)	25 instructions
Number of registers	32	16	**Typical power dissipation**	Not Available	1.5 W at 3.3V
Package Size	352 ball, 35mmX35mm	225 ball, 23mm on a side			

Table 2. Digital Signal Processors.

The processor chosen was the TMS320C6701 general-purpose digital signal processor manufactured by Texas Instruments (TI™). The reasons for this selection were many. From the outset of this research project, the intent was to choose a COTS device for the TMR design. The '6701 is a COTS device, single chip, Harvard Architecture, with a 32-bit data bus and a 19-bit address bus. The processor has many glueless interfaces to external devices therefore minimizing the need for additional off chip control logic. This latter feature is also true of the Analog Devices ADSP-21060.

The determining factor was the failure of ADSP-21060 in a radiation environment [Ref. 10]. While the'6701 has not yet been tested, pending funds Dr. James Anderson of

Anderson of the Massachusetts Institute of Technology will test it. The outcome of those tests will determine how far this project can proceed, because TID, SEB or destructive SEL are a function of the processor's manufacture and cannot be corrected with extra hardware or software. However, it should be noted that some radiation effects could be diminished with shielding, as with a piece of aluminum.

G. CHARARACTERISTICS OF THE SELECTED PROCESSOR

The TMS320C6701 is intended to provide high-speed floating-point signal processing. It is a modified Harvard Architecture device. It has an instruction set that is similar to a reduced-instruction-set-computer (RISC), which employs load/store architecture. It operates on low voltage (3.3 V) and has power-down capabilities to conserve power. It also has many glueless architecture features built into the processor to reduce the number of off-chip components. This decreases the size and power requirements for external ICs which would be required to implement the design. Some of the features of TMS320C6701 include: [Ref. 9]

- 8.3 or, 6.7 ns Instruction Cycle Time (selectable)
- Eight 32-Bit Instructions/Cycle
- 1 GFLOPS
- Very Long Instruction Word (VLIW) 'C67x CPU Core
- Eight Highly Independent Functional Units:
- Four ALUs (Floating- and Fixed-Point)
- Two ALUs (Fixed-Point)
- Two Multipliers (Floating- and Fixed-Point)
- Load-Store Architecture With 32 32-bit general-purpose registers
- Instruction Packing Reduces Code Size
- All Instructions Conditional
- Hardware Support for IEEE Single-Precision Instructions Floating-Point
- Hardware Support for IEEE Double-Precision Instructions Floating-Point
- Byte-Addressable (8-, 16-, 32-Bit Data)
- 512K-Bit Internal Program/Cache (16K 32-Bit Instructions)

- 512K-Bit Dual-Access Internal Data (64K Bytes)
- 32-Bit External Memory Interface (EMIF)
- Glueless Interface to Synchronous Memories: SDRAM and SBSRAM
- Glueless Interface to Asynchronous Memories: SRAM and EPROM
- 52M-Byte Addressable External Memory Space
- Four-Channel Bootloading
- Direct-Memory-Access (DMA) Controller with an Auxiliary Channel
- 16-Bit Host-Port Interface (HPI) with access to Entire Memory Map
- Two Multichannel Buffered Serial Ports (McBSPs)
- Up to 256 Channels Each
- Two 32-Bit General-Purpose Timers
- Flexible Phase-Locked-Loop (PLL) Clock Generator
- 352-Pin Ball Grid Array (BGA) Package (GJC Suffix)
- 0.18-mm/5-Level Metal Process
- CMOS Technology
- 3.3-V I/Os, 1.8-V Internal (120-, 150-MHz)

Figure 2 shows a block diagram of the TMS320C6701 DSP.

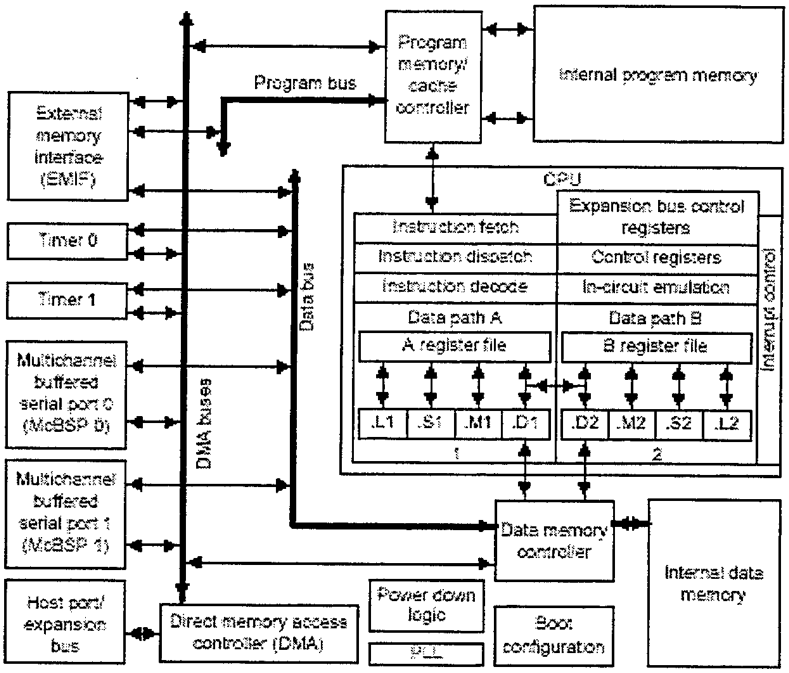

Figure 2. Block Diagram TMS320C6701 [From: Ref. 12].

1. The CPU

The CPU is a 32-bit RISC type floating/fix point execution engine, capable of executing one to eight instructions per clock cycle. It has a variable stage pipeline that can execute an instruction from a minimum of 7 stages to a maximum of 14 stages dependent upon the type of instruction. The CPU is divided into two execution paths, A and B. Each execution path contains functional units L, S, M, and D. It also includes 32 32-bit registers that are divided into two groups of 16 registers, one servicing data path A and the other servicing data path B [Ref. 12].

a. Pipeline Structure

All instructions flow through the fetch, decode, and execute stages of the pipeline. The fetch stage of the pipeline has four phases for all instructions, and the decode stage has two phases for all instructions. The execute stage of the pipeline requires a varying number of clock cycles, depending on the type of instruction. The stages for '6701 are show in Figure 3.

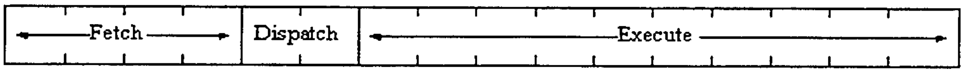

Figure 3. Pipeline Stages [From: Ref. 10].

(1) Program Fetch. Program fetch occupies the first 4 cycles of the pipeline, and consists of program address generation, program address send, program access ready wait, program fetch packet receive. It is important to note that the processor fetches eight instructions each time a program fetch cycle is initiated, so all fetch

addresses end with two zeros. Therefore if a program branches to a location that ends in other than double zero it will read the instructions prior to the branch location and discard the unnecessary instructions [Ref. 10].

(2) Instruction Dispatch. The instruction dispatch phase of the pipeline consists of two stages: the instruction dispatch and instruction decode. When a program is compiled, the program is broken into execution packets. An execution packet is a set of instructions that can be executed in parallel. To distinguish instructions that can be executed in parallel, a "1" will be placed in the last digit of the instruction, and up to eight instructions can be executed at once. Instruction dispatch is responsible for assigning an instruction to an execution unit. Again, the unit to be used is predetermined at compile time [Ref. 10].

(3) Execution. The execution phase can range from one to ten clock cycles, depending on what type of instruction is being executed. Parallel instructions entering the execution stage of the pipeline may complete at different clock cycles. Due to the different number of cycles for each instruction, there are no pipeline interlocks, and therefore any interdependencies must be taken care of at compile time [Ref. 10].

b. Register File

The general-purpose register file is broken into two groups, each with 16 32-bit registers. Group A registers service the A data path and B registers service the B data path. The '6701 general-purpose register files support data ranging in size from packed 16-bit data through 40-bit fixed-point and 64-bit floating-point data. Values larger than 32 bits, such as 40-bit long and 64-bit float quantities, are stored in register pairs. In these the 32 LSBs of data are placed in an even-numbered register and the remaining 8 or 32 MSBs in the next upper register (which is always an odd-numbered register) [Ref. 10].

c. *Functional Units*

The eight functional units in the ‘6701 data paths can be divided into two groups of four; each functional unit in one data path is almost identical to the corresponding unit in the other data path. The functional units are described in Table 3 [Ref. 10].

Functional Unit	Fixed-Point Operations	Floating-Point Operations
.L unit (.L1, .L2)	32/40-bit arithmetic and compare operations 32-bit logical operations Leftmost 1 or 0 counting for 32 bits Normalization count for 32 and 40 bits Byte shifts Data packing/unpacking 5-bit constant generation Dual 16-bit arithmetic operations Quad 8-bit arithmetic operations Dual 16-bit min/max operations Quad 8-bit min/max operations	Arithmetic operations DP SP, INT DP, INT SP conversion operations
.S unit (.S1, .S2)	32-bit arithmetic operations 32/40-bit shifts and 32-bit bit-field operations 32-bit logical operations Branches Constant generation Register transfers to/from control file (.S2 only) Byte shifts Data packing/unpacking Dual 16-bit compare operations Quad 8-bit compare operations Dual 16-bit shift operations Dual 16-bit saturated arithmetic operations Quad 8-bit saturated arithmetic operations	Compare SP DP conversion operations Absolute value operations Reciprocal and reciprocal square-operations
.M unit (.M1, .M2)	16 x 16 multiply operations 16 x 32 multiply operations Quad 8 x 8 multiply operations Dual 16 x 16 multiply with add/subtract operations Quad 8 x 8 multiply with add operation Bit expansion Bit interleaving/de-interleaving Variable shift operations Rotation Galois Field Multiply	32 X 32-bit fixed point multiply operations Floating-point multiply operations
.D unit (.D1, .D2)	32-bit add, subtract, linear and circular address calculation Loads and stores with 5-bit constant offset Loads and stores with 15-bit constant offset (.D2 only) Load and store double words with 5-constant Load and store double and non-aligned words 5-bit constant generation 32-bit logical operations	Load doubleword with 5-bit offset

Table 3. Function Units [From: Ref. 10].

2. Internal Program Control and Memory

The program memory controller, shown in Figure 2, provides the following functions: CPU and DMA requests to internal program memory and the required arbitration; CPU requests to external memory through the external memory interface (EMIF); and manages the internal program memory when it is configured as cache. The program memory/cache contains 64K bytes of RAM or, equivalently, 2K 256-bit fetch packets. The CPU, through the program memory controller, has a single-cycle throughput of 256-bit-wide connection to internal program memory. The memory/cache can be operated in four different modes, which are described in the paragraphs below.

a. Mapped Mode

In mapped mode, the program fetches from the internal program-memory-addresses and returns the fetch packet from that address. In the other modes, CPU accesses to this address range return undefined data. Mapped mode is the default state of the internal program memory at reset. The CPU cannot access internal program memory through the data memory controller [Ref. 10].

b. Cache Enabled

In cache-enabled mode, the first program fetch at an address causes a cache miss. In a cache miss, the fetch packet is loaded from the external memory interface (EMIF) and stored in the internal cache memory, one 32-bit instruction at a time. While the fetch packet is being loaded, the CPU is halted. Any subsequent read from a cached address causes a cache hit, and that fetch packet is sent to the CPU from the internal program memory without any wait states. Changing from mapped mode to cache enabled mode flushes the program cache. This mode transition is the only means of invalidating the cache [Ref. 10].

c. Cache Freeze

During a cache freeze, the cache retains its current state. A program read of a frozen cache is identical to a read of an enabled cache except that on a cache miss the data read from the external memory interface is not stored in the cache. Cache freeze ensures that critical program instructions are not overwritten in the cache [Ref. 10].

d. Cache Bypass

When the cache is bypassed, any program read fetches data from external memory. The program instructions are not stored in the cache memory [Ref. 10].

3. Data Memory Controller and Memory

As shown in Figure 2, the data memory controller connects the CPU and the direct memory access (DMA) controller to the internal data memory and performs the necessary arbitration; the DMA connects the CPU to the External Memory Interface Controller (EMIF) and to the on-chip peripherals through the peripheral bus controller. The data memory is 64K bytes on-chip memory to store data for quick access by the CPU. There are no wait states incurred when accessing it. It cannot be operated in a cache mode. It functions strictly like any other memory. The on-chip memory is organized into two blocks of 32K bytes. The DMA and CPU can access on-chip memory simultaneously as long as they access different blocks of memory [Ref. 10].

4. Direct Memory Access (DMA) Controller

The direct memory access (DMA) controller transfers data between regions in the memory map without the intervention of the CPU. The DMA controller allows movement of data to and from internal memory, internal peripherals, or external devices to occur in the background of CPU operation. The DMA controller has four independent programmable channels, allowing four different contexts for DMA operation. In addition, a fifth (auxiliary) channel allows the DMA controller to service requests from the host port interface (HPI). It is important to note that the DMA can operate independently of the CPU once it has been setup. For example, the DMA can move blocks of data from a first-in-first-out (FIFO) initiated by an internal or external interrupt to internal or external memory and then reset once the data transfer is complete, all without the intervention of the CPU [Ref. 10].

5. Peripheral Bus Controller

The peripherals are controlled by the CPU and the DMA controller through accesses of control registers. The CPU and the DMA controller access these registers through the peripheral data bus. The DMA controller directly accesses the peripheral bus controller, whereas the CPU accesses it through the data memory controller [Ref. 10].

6. Timers

The device has two 32-bit general-purpose timers that can be used to:

- Time events
- Count events
- Generate pulses

- Interrupt the CPU
- Send synchronization events to the DMA

The timers have two signaling modes and can be clocked by an internal or an external source. The timers have an input pin and an output pin. The input and output pins, (TINP and TOUT) can function as timer clock input and clock output. They can also be configured for general-purpose input and output, respectively [Ref. 10].

7. Interrupt Selector

The '6701 peripheral set has up to 32 interrupt sources. The CPU however has only12 interrupts available for its use. The interrupt selector allows the designer to choose and prioritize which 12 of the 32 the system needs to use. The interrupt selector also allows one to effectively change the polarity of external interrupt inputs. Table 4 includes a list of possible interrupt source that can be used to halt the processor [Ref. 10].

Interrupt Selection Number	Interrupt Acronym	Acronym Interrupt Description
00000b	DSPINT	Host processor to DSP interrupt
00001b	TINT0	Timer 0 interrupt
00010b	TINT1	Timer 1 interrupt
00011b	SD_INT	EMIF SDRAM timer interrupt
00100b	EXT_INT4	External interrupt pin 4
00101b	EXT_INT5	External interrupt pin 5
00110b	EXT_INT6	External interrupt pin 6
00111b	EXT_INT7	External interrupt pin 7
01000b	DMA_INT0	DMA channel 0 interrupt
01001b	DMA_INT1	DMA channel 1 interrupt
01010b	DMA_INT2	DMA channel 2 interrupt
01011b	DMA_INT3	DMA channel 3 interrupt
01100b	XINT0	McBSP 0 transmit interrupt
01101b	RINT0	McBSP 0 receive interrupt
01110b	XINT1	McBSP 1 transmit interrupt
01111b	RINT1	McBSP 1 receive interrupt
10000b		Reserved
10001b	XINT2	McBSP 2 transmit interrupt
10010b	RINT2	McBSP 2 receive interrupt
other		Reserved

Table 4. Interrupt Selection [From: Ref. 10].

8. Multi-Channel Buffer Serial Ports (McBSP)

The McBSP is based on the standard serial port interface devices. The McBSP provides:

- Full-duplex communication
- Double-buffered data registers, which allow a continuous data stream
- Independent framing and clocking for receive and transmit
- Direct interface to industry-standard codecs, analog interface chips and other serially connected A/D and D/A devices
- External shift clock or an internal, programmable frequency shift clock for data transfer
- Auto-buffering capability through the 5-channel DMA controller

The McBSP consists of a data path and a control path, which connect to external devices. Data is communicated to these external devices via separate pins for transmission and reception. Control information (clocking and frame synchronization) is communicated via four other pins. The device communicates to the McBSP via 32-bit-wide control registers accessible via the internal peripheral bus [Ref. 10].

9. External Memory Interface Controller (EMIF)

The EMIF controls the entire external memory interface and the rest of the '6701 including: the DMA controller, data memory controller, program memory controller, and the internal peripheral bus. The EMIF allows for glueless interface to external memory. The EMIF is controlled by a set of registers that are set during the booting up. These registers allow for maximum flexibility in selecting memory types and manufacturers [Ref. 10].

10. Host Port Interface (HPI)

The HPI allows for the DSP to be controlled by an external microprocessor. The '6701 allows the external processor full read/write access to its memory map. The HPI is a 16-bit wide interface that minimizes the bus width to minimize pinout. The HPI allow 32-bit data to be multiplexed over the 16-bit bus [Ref. 10].

I. SUMMARY

After completing a review of the processors the TMS320C6701 was selected as the choice for the fault tolerant design. The '6701 still requires testing in a radiation environment to ensure its reliability in regard to SEL, SEB and TID. The general architecture of the processor was discussed.

In the next chapter, some of the concepts of triple modular redundancy, a hardware technique, are covered.

THIS PAGE INTENTIONALLY LEFT BLANK

III. DETECTION AND CORRECTION OF SEU

A. OVERVIEW

As discussed in previous chapters, this thesis is concerned with SEU on the '6701 and not with the effects of TID, SEL and SEB. Those effects are generally destructive and cannot be corrected with hardware or software redundancy. They are only prevented by different design considerations and/or manufacturing techniques. So, it is evident that only bit flip (changing a binary to its opposite) or transient effects of SEU can be corrected with either software and/or hardware redundancy. In simplistic terms, the bit flip causes a change in the value of an instruction within the register, resulting in a flawed instruction being executed. The execution unit performs according to its instruction, but the instruction is wrong. In transient SEU, a correct instruction is given to the execution element, but this instruction or its data elements are corrupted by stray energy (a SEU event) within the execution element. This results in bad output, even though correct instruction and data elements were delivered. There is no functional way to discriminate between these two upsets, so for simplicity, they are addressed as one in the same. Because of the architecture of the '6701, it is necessary to take a detailed look at how an SEU might affect the chip's operation. In order to simplify the problem, transient effects during calculations will be grouped together with errors in registers. By studying the effects of SEU, we can perform detection, analysis, and correction of corrupted registers or transient SEU.

There are many ways to design a computing system so that it achieves some degree of fault tolerance when dealing with SEU. Fault tolerant computing systems employ combinations of hardware, software, time or information redundancy. Each has its strengths and weaknesses, but each can be categorized in one of two groups: time redundant or hardware redundant. Since there are radiation hardened DSP in the market already, it is necessary that the design proposed here remains as fast possible in order to justify choosing a non-radiation hardened processor. Therefore, this design will use triple modular redundancy (TMR) hardware to achieve fault tolerant design. The overall goal is to exploit all of the advances in the '6701 with minimum constraints on future design implementations. This overall goal includes the ability to use any memory type, the two serial ports, and control the '6701 by any external host.

B. TRIPLE MODULAR REDUNDANCY (TMR)

A common and simple idea for hardware redundancy is triple modular redundancy (TMR). Basically, triplicating the processing hardware allows the system to execute instructions in parallel and then majority vote the outputs to provide a correct system output as shown in Figure 4. For more information on this subject, see Reference 2 in the bibliography.

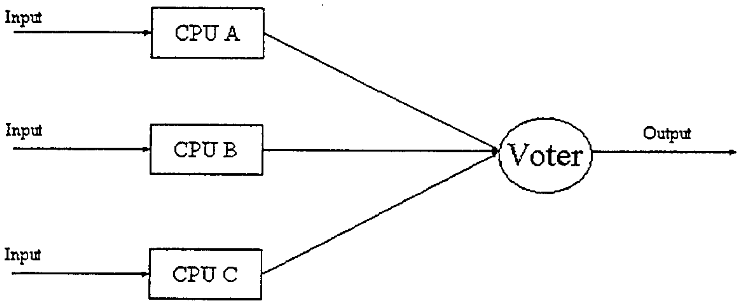

Figure 4. Basic TMR Circuit Implementation [From: Ref. 2].

Fundamentally in the TMR system, if one module becomes faulty, the other two modules mask the error of the faulty module through a majority vote.

In a TMR system with three digital signal processors, the most hazardous type of error is when an SEU causes one DSP to branch to an unexpected location. This causes the system to go out of synchronization. The affected processor will continue to operate out of sync with the other two processors until all three processors are reset to the same location in the execution cycle. Until the faulty unit is reset, the system is no longer TMR system. It becomes a dual processor system that can detect errors but cannot correct errors. Since this is a fault tolerant design, this is an unacceptable state of operation.

One of the primary disadvantages of a TMR system is that it is no more reliable than the voter. The voter is now a single point of failure and is the weakest link in the chain. A solution to this process would be to triplicate the voter also, but how would one select the system output among the three outputs? You could then vote the outputs of the three voters but then second level voter becomes the single point of failure. Since triplicated voters still require a final vote to be made by one voter, this idea is discarded because it just adds another layer of logic delay and more power. Since there is no practical advantage for the redundancy described above, a single stage TMR system is selected for this project. This then poses extra reliability requirements for the voter.

1. Voting Techniques

There are several ways a TMR system can be designed. It can be designed with three completely independent systems, which then vote their outputs. Each system contains its own sensors and digital signal processor, which calculate their output. This output is then sent to a voter. Using this design presents a problem though. Is the processor at fault or is there a slight difference of quantization by the analog to digital converters? The slight difference in quantization would masquerade as an error, when in fact there is little difference among the processors' results.

A second design might use three processors working in parallel that have the same inputs and then vote the results. This technique ensures each processor is working with the same data. They should all output identical results. Of course, this is another point at which a single point failure could cripple the system. Since it is nearly impossible to build identical analog to digital converters, a single input shall drive all three processors to ensure that each is starting with the identical data. The three processors should then result in three identical outputs. The output will then be voted upon to determine if an error has occurred.

The process of voting digital data itself is rather simple. The three outputs from the DSP are used to find a majority. The same process can also be used to detect when an error occurs. Figure 5 shows how error detection and error correction can occur simultaneously. The three AND gates and one OR gate correct for errors while the AND, NOR, and OR gates are used to determine if an error has occurred.

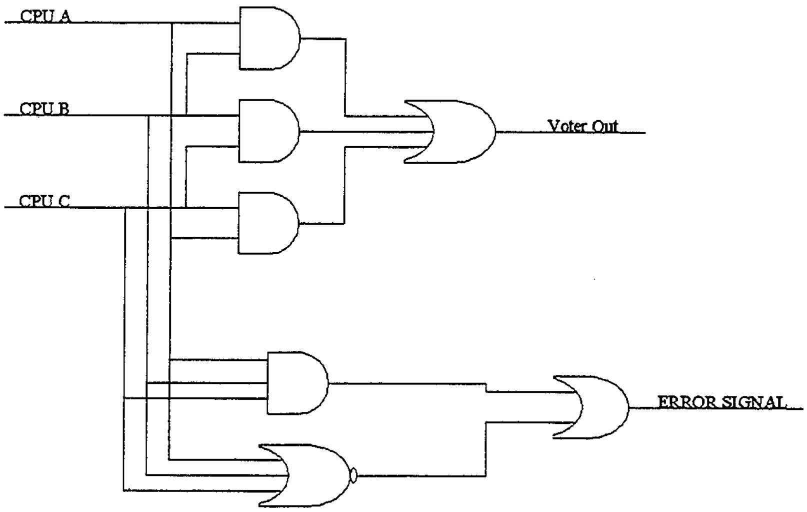

Figure 5. Majority-Voter and Error Detection.

2. Voting Issues

Voting timing becomes a major issue when designing with a microprocessor and its outputs. Since the voting is performed on all outputs, a careful timing analysis must be performed to answer the following questions: when do you vote? What impact does it have on the memory cycle? What impact does it have on bi-directional busses? Can the three processors be synchronized? How do you stop the processors once an error has occurred? How do you find identical errors?

The answers to these questions are dependent upon the type of microprocessor selected. Since I have already selected the '6701, I will answer these questions as they pertain only to the '6701.

First let us examine the CE, BE, data and address busses to determine when we could vote the busses and how voting affects the memory bus cycle. Different memory types affect how the memory bus is operated. The memory types must be individually analyzed due to their unique operation of the memory bus. In the following sections, each memory type will be analyzed.

a. Asynchronous Peripheral

In order to assure proper timing, a review of the memory cycle is necessary. Asynchronous peripherals is a term that refers to a vast spectrum of memory types including such familiar types as asynchronous random access memory (ASRAM), flash memory, read only memory (ROM), first-in-first-out (FIFO) buffers, dynamic random access memory (DRAM) and many more peripheral devices. The unique glueless interface of the '6701 allows for a single type of memory to occupy an entire CE memory space without any extra logic to generate chip-select. Since there are only four CE memory spaces, the asynchronous peripherals are usually grouped together in one memory space. Therefore, external logic devices are required to generate chip-select signal, if more than one type of memory is located within one CE space. An internal control register, which is initialized during the boot cycle, controls each CE space (Figure 6). When the CE control register is set for an asynchronous memory type (putting the appropriate code in the Mtype field), it allows for manipulation of the timing of the read and write cycles.

EMIF CE(0/1/2/3) Space Control Register

31 28	27 22	21 20	19 16
Write Setup	Write Strobe	Write Hold	Read Setup

15 14	13 8	7	6 4	3 2	1 0
Rsvd	Read Strobe	Rsvd	Mtype	Rsvd	Read Hold

Figure 6. EMIF Control Register [From: Ref. 11].

By setting the CE fields that control write-setup, write-strobe and write-hold, the '6701 inserts the required number of wait-states. In the '6701, the ability exists to operate seamlessly with external memory devices. This feature does not need the normal acknowledgment signal from the memory device to the CPU to indicate that it is ready for a write. By eliminating the acknowledgment signal, the memory cycle is sped up. The same is also true for the read cycle. This feature can be exploited by inserting a required amount of time in order to perform the voting of data and address busses.

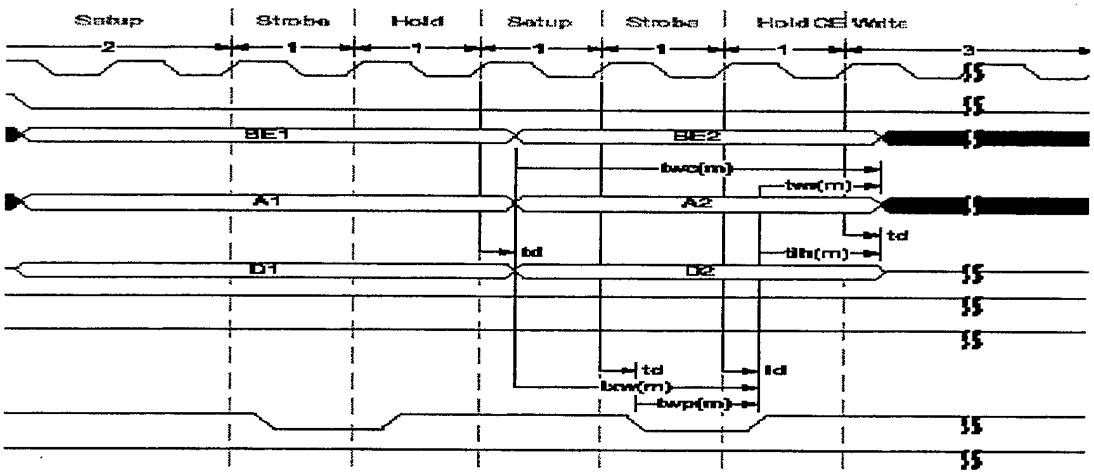

Figure 7. Asynchronous Write Cycle [From: Ref. 13].

From the write cycle (Figure 7), it is observed that address, CE, BE and data are all asserted at the same time. The logical place to insert the required wait states is in the setup time. Because the data bus is a bi-directional bus, there is a need to ensure that no stray signals are generated. This implies a need for a tri-state device. Tri-state devices require a control signal in order to be activated. For the write cycle, the write strobe could be used to control the flow of data to memory.

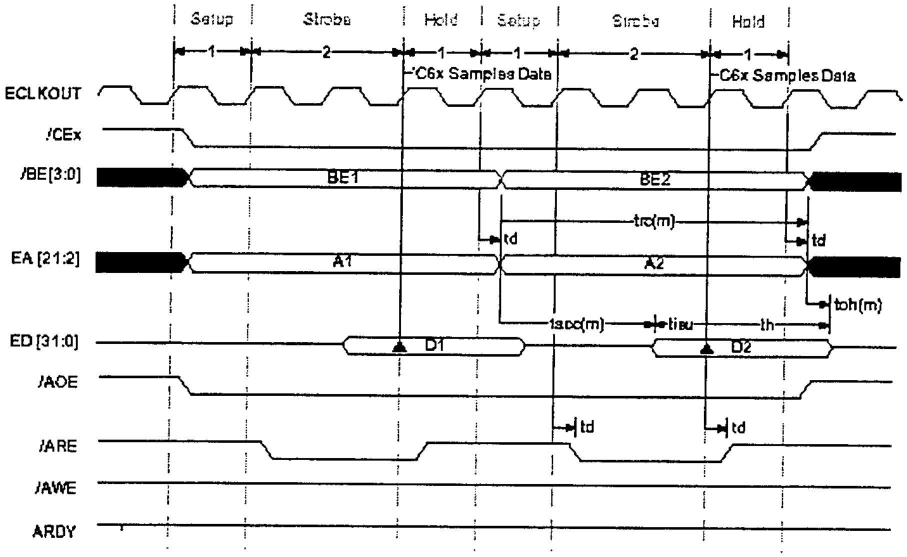

Figure 8. Asynchronous Read Cycle [From: Ref. 13].

From the read cycle (Figure 8), it is observed that address, CE, BE and data are all asserted at the same time. Again, the flow of data must be controlled. The '/ARE' signal could be used as the control signal.

For both the read and write cycles, the address, BE, and CE signals can be voted upon without the need for a control signal to regulate the flow to memory. By allowing the signals to propagate through the voting circuit without a control signal, the memory maybe operated at a higher speed. But generating an error signal to interrupt the processor requires that the /ARE and the /AWE signals are used to control exactly when the error generator is sampled. The use of these two signals ensures a sufficient amount of settlement time passes before the vote is sampled. This also allows for a small amount of skew between the processors without detriment.

b. Synchronous Burst Random Access Memory (SBRAM)

Like the asynchronous memory types, a detailed look at the read/write cycle is needed to ensure the proper operation of the bi-directional data bus. Synchronous Burst Random Access Memory (SBRAM) is normally ideal for DSP because of its high-speed operation, which allows for rapid transfer of data to and from memory. Unlike the asynchronous memory, the CE memory space control has a more limited amount of control over the memory cycle. Operation at full CPU clock speed or half CPU clock speed is the only control over SBSRAM. Because both the timer and external interrupt inputs require a minimum hold time of two clock cycles, SBSRAM can only be operated at half CPU clock speed. This memory type may not be used in the final application, but it is included as an option if the designer desires it.

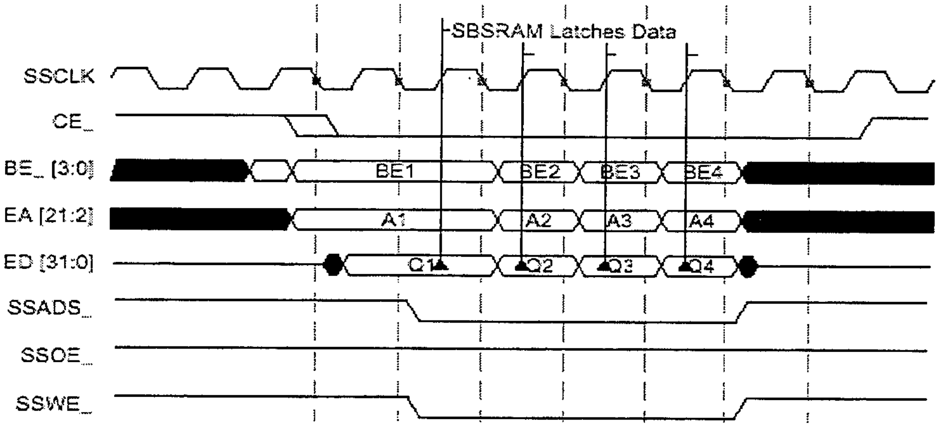

Figure 9. SBSRAM Write cycle [From: Ref. 14].

The SBSRAM write cycle, shown in Figure 9, has an initial two-memory clock-cycle delay setup time before the first write occurs. A maximum burst length of four write locations can be performed during any burst cycle. Like asynchronous memory, the write strobe (SSWE) can be used to control the flow of data on the data bus from the CPU to the SBSRAM. There is no need to worry about triggering the voting error signal for each address location and data set, because the external interrupt and timing register require a minimum hold of 2 internal clock cycles. Therefore, a small perturbation, caused by slight clock skew between processors or transitions of address/data lines, will not cause false positives.

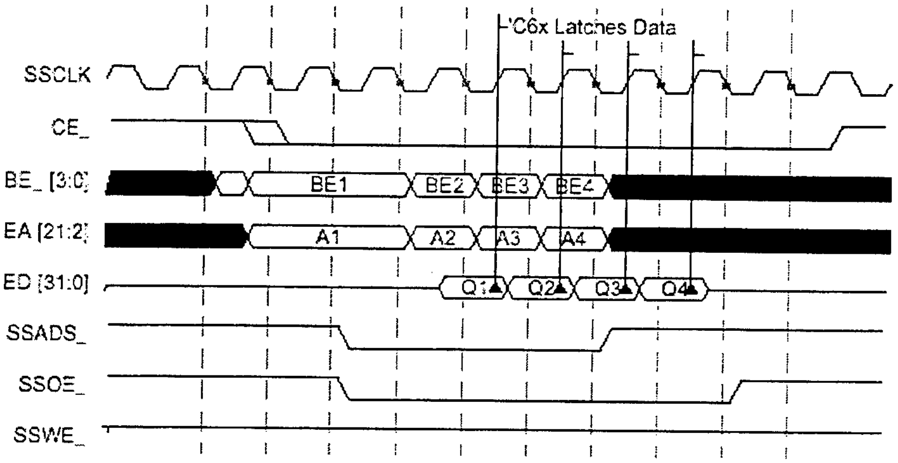

Figure 10. SBRAM Read Cycle [From: Ref. 14].

The SBSRAM read cycle, shown in Figure 10, also has an initial two memory-cycles before the first read is latched into the CPU. The SSOE signal can be used to control the data bus flow from memory to the CPU. Like the write cycle, any minor perturbations in voting of address lines will be ignored by the CPU due to the minimum of two CPU clock-cycle hold times.

The CE, BE and address lines can be handled the same as the asynchronous memory types. The most significant factor in using SBRAM in this design will be the absolute speed of the voting circuit. The entire memory cycle cannot exceed two internal clock-cycles (operating at the slowest speed of 133 MHz), which is a maximum of 15 nsec. In order to achieve these results, the read (SSOE) and write

(SSWE) will only be voted upon and will not factor into the generation of the error signal to interrupt the CPU.

c. *Synchronous Dynamic Random Access Memory (SDRAM)*

Like SBSRAM, Synchronous Dynamic Random Access Memory (SDRAM) is ideal for DSP because of its high-speed access times. SDRAM also would have the added benefit of lower power requirements. As with the other two memory types the data bus must maintain a tri-state functionality. Figure 11 and 12 show the memory cycle for SDRAM type memory architecture. The problem revealed by the two timing diagrams is that there is no signal that can be used to indicate a read cycle is taking place and that the bypass buffer should be turned on. One could use the negative of the write cycle (/SDWE), but it then would be turned on all the time regardless whether a read cycle is occurring or not. This would be a suitable solution as long as no other types of memory share a common data bus, but to keep the design simple and to avoid building an additional memory bus, SDRAM was rejected for this project.

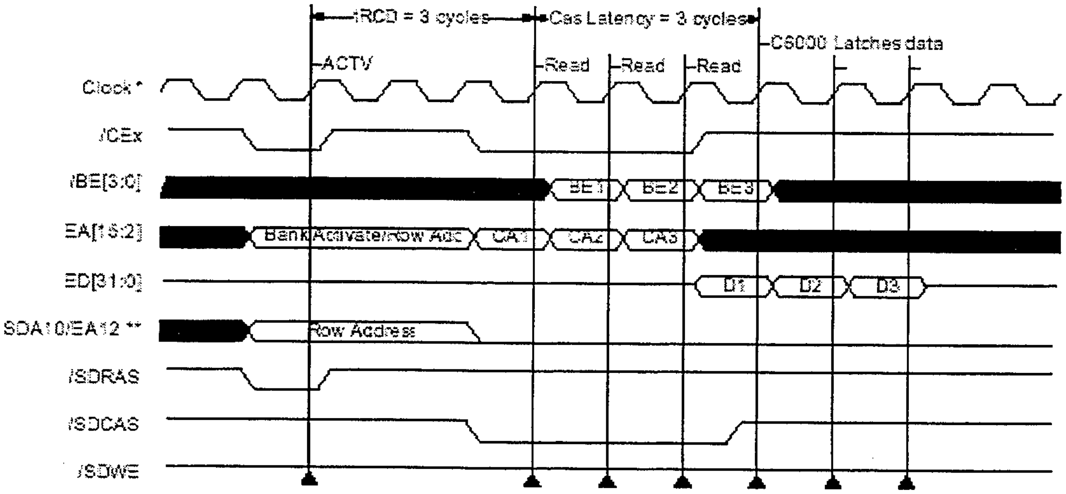

Figure 11. '6701 SDRAM Read Clock Cycle [From: Ref. 16].

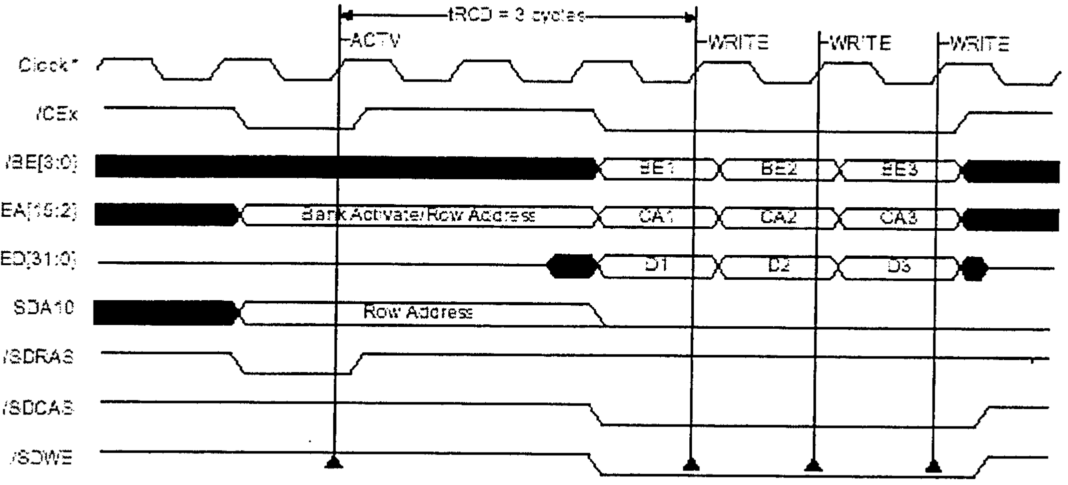

Figure 12. '6701 SDRAM Read Cycle [From: Ref. 16].

d. Miscellaneous Timing Issues

Processor synchronization depends upon each CPU having nearly identical clock input signals and each processor being reset at the same time. The use of a single clock generator for all three processors is important. The frequency of the clock input to the CPUs is one half the internal clock rate.

There is an additional point of concern that is not necessarily central to the thesis, but deserves mentioning. Identical errors will not be detectable by this hardware implementation. but it is highly unlikely that identical errors across 19 address lines, 4 BE lines, 4 CE lines and 32 data lines would occur. There are three scenarios that can be described: error in one processor, and identical errors in two processors, and non-identical errors in two or more processors. The first case, a unique error in one processor, is both detectable and recoverable and is the focus of this work. The second case, identical errors in two processors, is not detectable and not recoverable. The third scenario, non-identical errors in two or more processors, is detectable and correctable.

The outputs from the HPI are not voted upon, because the HPI will be used to extract information from the processors to determine where the error occurred. When an error signal occurs, the processors will go into an interrupt service routine that will save all registers to internal memory. This will preserve the fault and allow an external-host processor to read the internal memory of each processor, so that it may be determined which was at fault and which register was corrupted. After saving the registers internally, the processors write all the registers through the voter to an external memory location. By writing to the external memory, the processor in fault will be masked out by the other two. The corrected data can then be read back into the appropriate registers and the CPUs can resume their functions. This re-synchronizes the CPUs.

C. ANALYSIS OF SEU ON TMS320C6701

As discussed in the previous section, the TMR design can detect only when an error is written to an external device or an unexpected read or write occurs. In this section, each sub-section of the ‘6701 will be examined in detail, specifically for the affects of radiation induced SEU. It must be determined if an SEU can be detected and how it will affect the processor. It is assumed that any transient effects caused during calculations will appear to be the same as if the original register(s) being operated upon was (were) corrupted. In other words, it may look corrupted, when it is actually not corrupted.

1. Error in Program RAM or Cache

If a bit-flip occurs in the program RAM or cache, an error may be detected in the data or address busses. Pin pointing this error will be unlikely, unless program memory is operated in memory-mapped mode. It can be corrected in an interrupt handling routine that changes program memory from operating in cache-mode to memory-mapped mode and then back to cache. This will flush the cache and each processor will reload its cache.

2. Error in Internal Data Memory

An error in the data memory will only be detected when a vote-error is detected on the data bus. Since internal memory is used for storage of constants that are used in FFTs, FIR filters and IIR filters, it becomes critical that these values are not corrupted. This is a very important issue if numerous results are stored to internal memory before a result is exported to an external device/memory. A solution to this problem is to periodically update the constants using one of the DMA channels. By using a DMA channel, minimum impact on the processor throughput can be maintained while improving the reliability.

3. SEU Error Induced into Registers

The registers of a CPU are critical to its correct performance. If these registers become corrupted by an SEU, it is necessary to determine what the repercussions of the corrupted register might be and how a faulty register is detected. Appendix A takes a detailed look at all of the control registers of the '6701. There are four main categories, into which all errors fall: detectable address errors, detectable data errors, undetectable errors and undefined errors. The detectable address and data errors can easily be dealt with by the TMR design. Undetectable errors present a challenge to this process. These undetectable errors occur when reading/writing to internal memory. Most of the undetectable errors will be eventually detected. The detection will occur during writing data to external devices. The classification of undefined errors is exclusively reserved for SEU occurring in reserved fields in control registers. Without detailed documentation from Texas Instruments, it is impossible to determine how the '6701 will behave when a reserve field is corrupted.

D. SPECIAL CONSIDERATIONS FOR DSP

Due to the nature of signal processing, data errors are not as important as data errors in other digital processing systems (i.e. general-purpose microprocessors). DSP is an approximation of the real world events using FIR filters, IIR filters and other types of signal processing techniques to reconstruct or model the original signal. In addition, nearly all digitally processed signals will be converted back to the analog world. Therefore, stray perturbations will be smoothed out.

For example, if a sound signal is digital processed and recorded that has a few "spikes of high frequency interference" introduced by the recording process, when the recorded signal is played back, the amplifier-speaker system can only respond so quickly (the system acts like a low-pass filter). The high frequency interference will be attenuated by the system response characteristics. Therefore, even large errors will be minimized making data errors in DSP less important than data errors in general purpose processors.

1. Address Errors

Data and address errors are not unique to DSP; they happen to all non-radhard devices in a space environment. A more pressing issue is how to handle DSP address errors. They must be handled in the same way as any other microprocessor.

Address errors may branch to an undefined location in memory. They could branch to the middle of a sub-program. In any case, it is unexpected and will quickly result in system failure. Due to this system criticality, address errors must be handled immediately by system. This is accomplished when the voter generates an interrupt signal to the CPU. Then, the interrupt routine can effectively handle the error. This interrupt handling routine is discussed in Chapter IV.

2. Data Errors

Data errors can be handled differently because of the nature of DSP, as we discovered in the preceding paragraphs. Sometimes an error occurs early in the processor history and remains undiscovered until much later in the process. By then, it would be pointless to interrupt the processors. For example, while performing an FFT, internal data memory is often used to store intermediate values to speed the calculations. An error could be present there. This error would then carry all the way through to the final FFT output. Suppose that you have 32-bits per point and 1024 points and you want to perform an FFT. It requires approximately 40,000 operations; all of which is done to output 1024 32-bit words. A few of these words at the end could have errors, but to halt the processor at this time would be inefficient, and since DSP usually convert the end product into analog anyway, these errors become overtaken by events.

An important, overall assumption to this thesis has three component assumptions. The first is that SEUs occur infrequently in the '6701, even in a space environment. The second is that the voting process masks out all single processor failures. The third component assumption is that coincidental processor failures are very infrequent. Based on the previous paragraphs, therefore, it becomes apparent why data errors are treated differently than address errors.

The data voter will generate pulses to be counted by each processor. An interrupt will occur when there are an excessive number of data errors. By counting the errors and interrupting the processor when excessive number of data errors occurs, the probability of multiple processors being in error at once is limited. What constitutes an excessive number of errors is left for latter work.

3. Variable Pipe Length

Another reason to service address errors immediately, beyond the fact that branching to undefined locations is undesirable, is the variable pipeline length. By not allowing address errors to persist, the chance for a processor branching to an undefined location can be minimized. A type of undetectable error occurs when instructions write results to internal memory locations. These errors will only be detected once they are written to external memory location. Therefore, programs using internal memory locations should periodically scrub the internal data memory. Good computer science dictates that internal data memory should only be used for intermediate calculations. Anything that requires long-term storage should be stored off chip. This ensures agreement between processors and provides for better system reliability.

E. SUMMARY

In summary, Chapter III looked at some of the affects of radiation on the '6701 and how to deal with SEU. A TMR design was proposed as a way to deal with SEU. The implementation of '6701 TMR design was also examined. Alternative schemes were discussed for the handling of address and data errors, as well as the implications of when, how and to what extent to mitigate them. This chapter leads naturally to a discussion of specific implementation issues. Chapter IV will deal with specific implementation of the '6701 in a TMR design.

IV. TMR TESTBED DESIGN

A. OVERVIEW

In order to observe the performance and behavior of a digital signal processor in the presents of radiation induced SEUs, the address, data and control busses must be monitored. In a digital signal processor, there is not an efficient built-in mechanism to indicate to external devices and/or observers that an SEU induced error have occurred. SEU induced errors may cause the processor to “lock up” or “crash,” which is detectable, but is of little use when trying to trouble-shoot or monitor the performance of the system [Ref. 2].

Monitoring of the address and data busses presents another problem. Without a separate entity which is deemed, or assumed, to be error free there is not a way to tell if the information that appears on the busses is error free or not. In addition, in the presence of radiation induced SEUs, the ability to correct such faults once detected are a necessary characteristic.

In this testbed design, triple modular redundancy (TMR) was chosen to allow monitoring of three identical processors running identical programs. The majority voting used in conjunction with TMR provides detection of an SEU that has been manifested as a disagreement among the address, data, and control busses of the three digital signal processors. The majority voter also allows the masking and thus correction of these SEU induced disagreements. The address, data, and control bus information from the two digital signal processors that agree is used to start, control, and complete each bus cycle.

As explained in Chapter III, this argument assumes that identical faults, or errors, will not occur in two different microprocessors and produce the same erroneous results on their associated busses. If this should occur, then the majority would be in an error state and we propose that this is unlikely.

B. TRIPLE MODULAR REDUNDANT DIGITAL SIGNAL PROCESSOR

Having reviewed the concept of TMR, what follows is a description on how they might be employed with three digital signal processors. Building the testbed using the TI™ TMS320C6701 floating-point digital signal makes it useful to examine what is necessary in constructing a board with three '6701's operating in a TMR design.

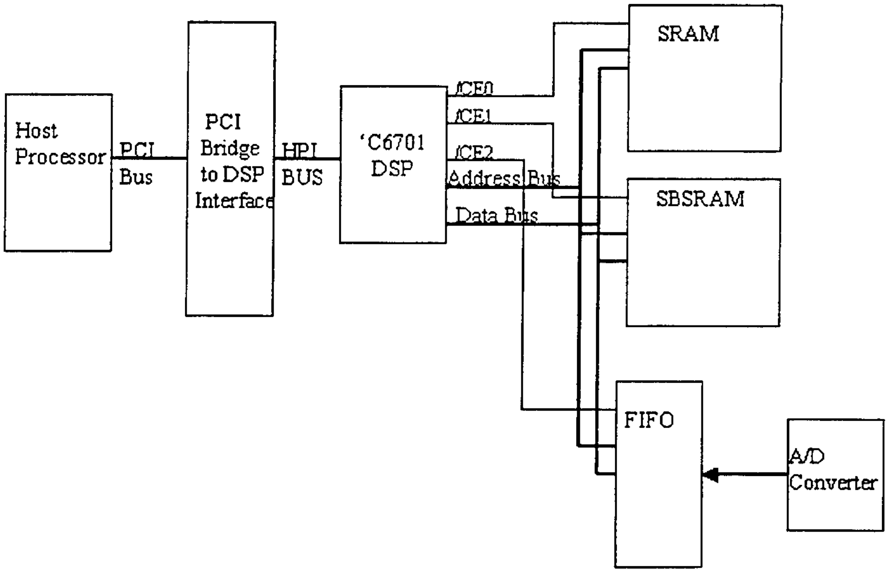

Figure 13. Simple '6701 Board Design.

Figure 13 is a simple non-specific digital signal processing board design. The final design, to be used by an application, will need to incorporate the overall requirements of a DSP system to determine what components are necessary. For example, the FIFO in Figure 13 is used to import data from an analog-to-digital (A/D) converter. However, many applications have a serial bus interface, which would require elimination of the FIFO and the use of a multi-channel serial bus interface (McBSP). In addition to parallel/serial bus interface, the type of DSP processing application directly affects the memory types and memory space required to perform image processing in comparison with frequency spectrum analysis of signals.

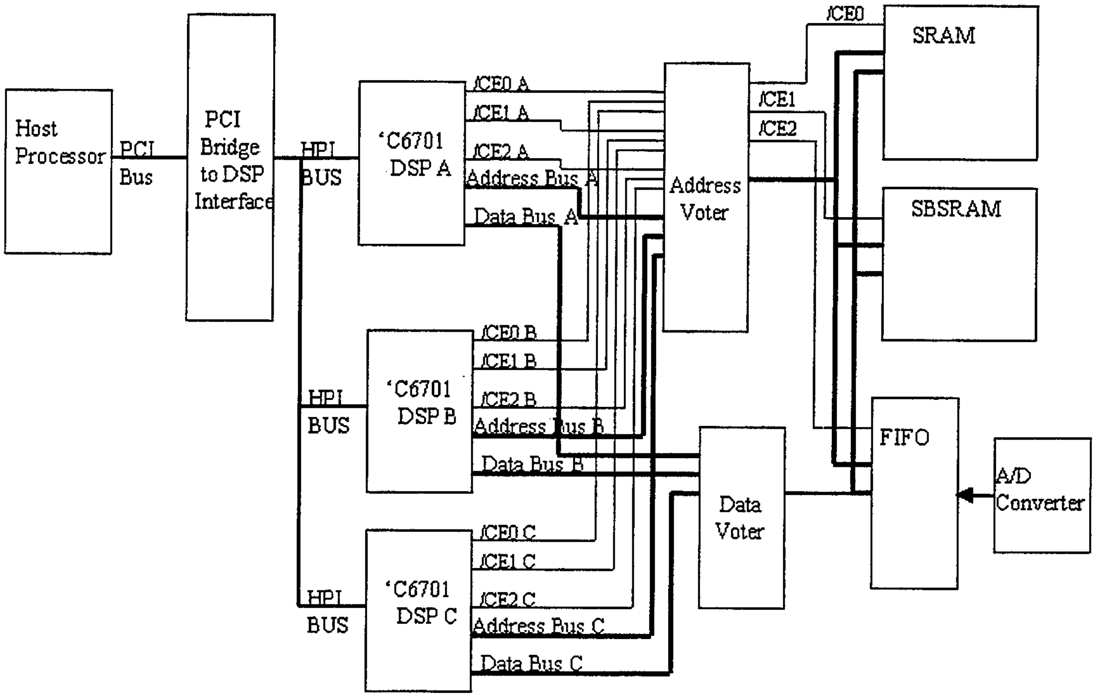

Figure 14. TMR '6701 Board Design.

Expanding on this simple system, Figure 14 shows a block diagram of a TMR system using '6701 digital-signal-processor. Figure 14 also shows the additional hardware blocks necessary to implement majority voting of the address, data, and CE busses.

A significant issue when using three processors in a TMR design is the synchronization of the processors. The processors will have one clock generator input that will drive all three '6701's. As discussed in Chapter III, the address bus and CE signals will be voted with simple logic devices and will not be clocked. This design will act as a propagation delay only, but to ensure the address and CE voters do not generate false error signals, a control buffer is placed at the end of the error generation circuit. The control signal will be a combination of the read/write strobes. The strobes come sufficiently late in the read/write cycle to allow for slight clock skew between processors. The voted address and CE signals do not have control buffers to impede flow of address and CE signals to external memory locations. A detailed look at this design will be presented in following sections.

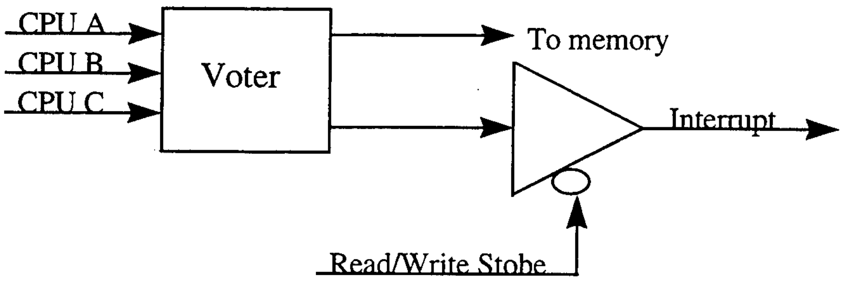

Figure 15. Interrupt Buffer Control Circuit.

The data bus synchronization requires more design considerations. Since the data bus is a tri-state bus and the voting is only occurring in one direction, care must be taken to ensure proper bus operation. The tri-state design must be maintained to ensure only one driver is driving the bus. Introducing the voter creates the possibility of multiple devices driving the bus. Therefore, a buffer must be placed at the end of voting circuit and a buffer must be placed in the bypass path, bypassing the voter, for data to travel from the memory to the processor. As in the case of the address voter, the read and write strobes can be used to control these buffers. In addition, a buffer to allow a skew margin between processors will control the error signal generated by the data voter. A detailed look at this design will be presented in following sections.

1. Testbed Operation Summary

The testbed contains three TI™ TMS320C6701 digital signal processors, operating at 120 MHz, executing the same program and service routine. Each will use internal SRAM memory to capture critical control-registers, address and data bus information during an interrupt handling routine. There is one PCI-to-HPI bridge to control the digital signal processor. The host microprocessor, on the other side of the PCI-to-HPI Bridge, is responsible for writing the execution program instructions into the memory map of the three digital signal processors. The host processor will also be used to extract the fault data during interrupts by reading each digital signal processor's internal memory. The host processor will access each '6701 independently to determine which processor is affected by an SEU event. All of this is accomplished by using the feature of a HPI. The '6701 is designed for an external processor to have control over the entire memory map of the '6701. The data traveling over the HPI bus will not be voted.

The address busses, data busses and control signals will be majority voted and combine the outputs from the three processors. These voted busses are then used to access the same SRAM, SBRAM and FIFO. The FIFO will be used as a parallel bus interface between the digital signal processors and an analog-to-digital (A/D) converter. Excluding the A/D converter will allow for follow-on designers to choose the type of A/D converter that is best suited for their application.

Address and data errors will be handled separately. Address errors will generate an interrupt signal to each processor. Then each digital signal processor will enter identical interrupt handling routine. Data errors will generate a signal pulse that will be sent to each processor's internal timer where it will be counted. The internal timer is a 32-bit register that can count from 0 to 2^{32}-1. It also can be used as a down counter and can be set to interrupt at any desired number of pulsing events. The reason for handling for data and address errors separately has already been discussed in Chapter III.

The following sections will discuss normal operations, error detection, error prevention, and error correction. Each will look at how they impact the systems operation.

a. Normal (Error free) Operations

At the beginning of a bus cycle (read or write), the address, BE and CE control signals are voted. The two levels of logic required to majority vote will only delay the address, BE and CE signals from arriving at the memory location. This delay can be incorporated into the setup time required by the memory.

The data bus must be maintained slightly differently than the address bus. Because the data bus is a tri-state bus, a control signal must be used to ensure there is only one driver driving the bus. Therefore, at the conclusion of the majority voting a control buffer is placed to ensure forward direct (write) control. A bypass buffer, bypassing the voter circuit, is placed in the flow of data from memory to the processors. The write and read strobes will be used to control flow of data.

The write and read strobes will be voted upon, but they will not be used in generating error signals to the processors. Since the likelihood of identical errors is considered small, it will be assumed that the chance of two processors generating identical read/write strobe errors is very small.

b. Error Detection

The error detection circuit has a minimum of three levels of logic, but since it requires "ORing" either 32 error signals (data bus), or 27 error signals (address bus, BE and CE signals) it also requires several levels of logic. Although the voting circuit in a programmable logic device, it was assumed that it could be done with four levels of logic, two layers of logic for each bit to generate an error signal then followed by 4 8-bit OR gates followed by one 4-bit OR gate to generate the final interrupt or error pulse signals. The final error signals will require a control buffer to prevent slight variations in arrival times from generating errors.

Address errors will cause an interrupt signal to the processor. The current bus cycle will be completed and then the processors will halt and start an interrupt handling routine. The error will be masked by the voter during read and write operations through the majority voter.

Data errors will cause an error pulse to be sent to the timers on each signal processor to be counted. When enough data error signals have been generated, the processors will enter the same interrupt handling routine as an address error-interrupt handler. Using the internal counter provides flexibility to help determine an optimum number of errors before an interrupt. The number of errors can be balanced against system accuracy and maintaining a high data throughput. For example, if a processor is bogged down doing numerous interrupt-handling routines, it will eventually cause the input FIFO to overflow because the processor cannot service the FIFO. The loss of data due to the interrupt handler will have the same affect as if a processor had corrupted the data.

c. Error Prevention

Using the internal data memory to store coefficients required to perform FFT, FIR filters and other DSP algorithms is an important assumption in this design. To limit the possibility of these coefficients becoming corrupted, they should be periodically refreshed. The refreshing routine will have limited impact on the processor's throughput because a DMA channel will be used to perform this refresh in the background of the signal processor's CPU operations. The same procedures should also be applied to any other values that are used repetitively and are stored in internal data memory. This implementation will require modification and/or development of special code that can be used by the operating system and/or application specific program.

d. Error Correction

Upon receipt of an interrupt, either from an address interrupt or timer interrupt, each processor executes the same interrupt service routine. The beginning of the routine will mask out all other interrupts. The internal general-purpose registers and configuration registers are written to a reserved internal data memory location. In addition to writing the registers to internal memory, they will also be written to a dedicated external memory location. By writing the data to an external memory location, a majority vote can be performed, thus masking out the corrupted data. The processors can then read back this information and be reset to a known and corrected state. The "faulty" processor will now have been "corrected" and re-synchronized. In addition to reading back the corrected registers, coefficients used for algorithms need to be read from their external memory locations and rewritten to internal data memory. To ensure the program cache is not corrupted, it should be invalidated. The simple procedure of switching from cache mode to memory-mapped mode and back to cache mode will effectively invalidate the program cache.

e. Error Monitoring

Since error monitoring is an important aspect of this design, an external host processor has been added which can be used to perform this error analysis. Since the external host processor has full access to the memory map of each processor individually, it can then read the internal data memories. The dedicated internal data memory will contain the information on where and possibly how the error occurred.

2. TI™ TMS320C6701 Simulation

A model for the complete '6701 digital signal processor is not available for simulation of the testbed design. Therefore, in order to develop the concept for this design, a detailed analysis was performed on the read/write cycle to ensure proper behavior of the memory system. One assumption that was used was the delay incurred for the voting logic [Ref. 3 and 6]. From previous design projects, a timing delay of seven nanoseconds was used as the delay incurred when generating a voted signal. The implemented circuits were placed on a Xilinks™ programmable chip. In addition to the timing requirements for the voting circuit, arbitrary components were also used in the design. These assumptions do not undermine the validity of this thesis since it is a demonstration project. Specific components were not selected due to the rapidly changing component market and the required testing of each component in a radiation environment.

In the description of the blocks and in associated figures, the following convention has been used. Signal and bus names which are bold and italicized, such as ***FORCE_A***, are intend to match the same signal and bus names in the overall schematic in Appendix B. This should make them easy to cross-reference. In addition, bus names that have an underscore followed by a letter and then end with a series of numbers, such as ***AD_A<21..2>***, represent bus signals that come from each of the three signal processors.

C. TI™ TMS320C6701 ADDRESS, BE, AND CE BUS INTERFACE

In this section, we will demonstrate that the address (***EA<21:2>***), ***BE,*** and ***CE*** busses meet the manufacturers specifications for the '6701.

The peripheral user's manual for the '6701 digital signal processor [Ref. 10] was used in conjunction with the ASRAM and SBSRAM implementation guides [Refs. 12 and 13]. Generic data sheets were used for the memory, which demonstrates that the timing requirements meet manufacturers' specifications.

1. Memory Read/Write Cycle

The read and write cycles were lumped together because there is no difference between the two. The ***CE*** signal is the most important of the three signals. The ***CE*** signal is connected to the chip enable pin on memory chips and other peripheral devices.

The chip enable signal enables the operation of memory and other peripheral devices. Using this knowledge and the fact that the ***EA*** and ***BE*** busses are unidirectional, a simple voting circuit can be directly implemented into the bus architectures. Each line will be voted to produce a system output (***V_EA<21:2>, V_BE<3:0>*** and ***V_CE<3:0>***) and generate an error signal (***EA_ERROR***). To generate ***V_EA<21:2>, V_BE<3:0>*** and ***V_CE<3:0>*** a majority voter and error generator was used using the inputs from each processor corresponding signals. The delay in generating the combined system output was treated as a propagation delay that could be easily added into memory timing requirement. Figures 16 and 17 demonstrate the abilities to use asynchronous memory types and the synchronous burst SRAM.

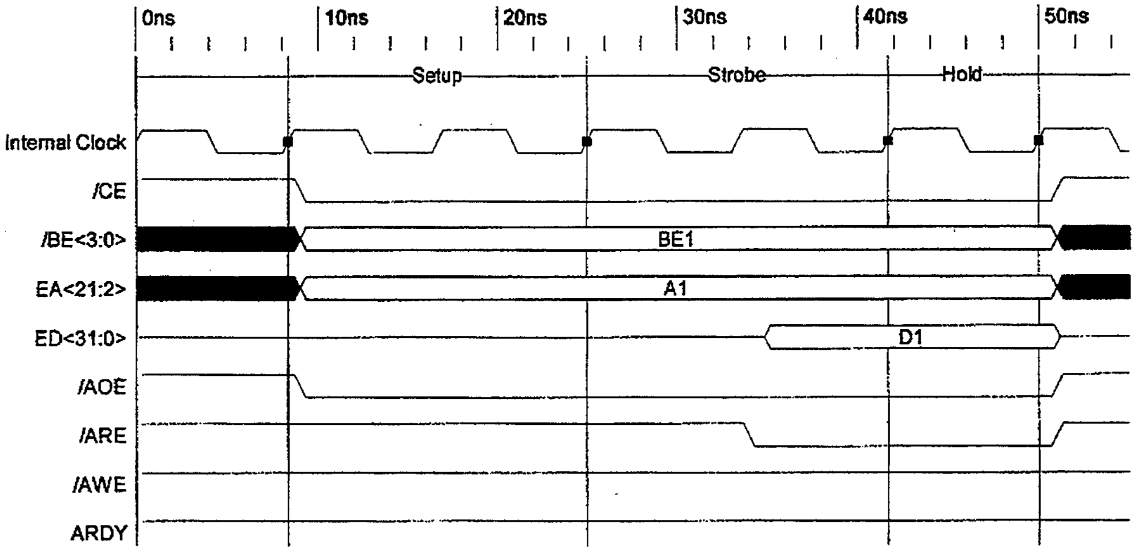

Figure 16. Asynchronous SRAM Write Cycle with Voting Delay.

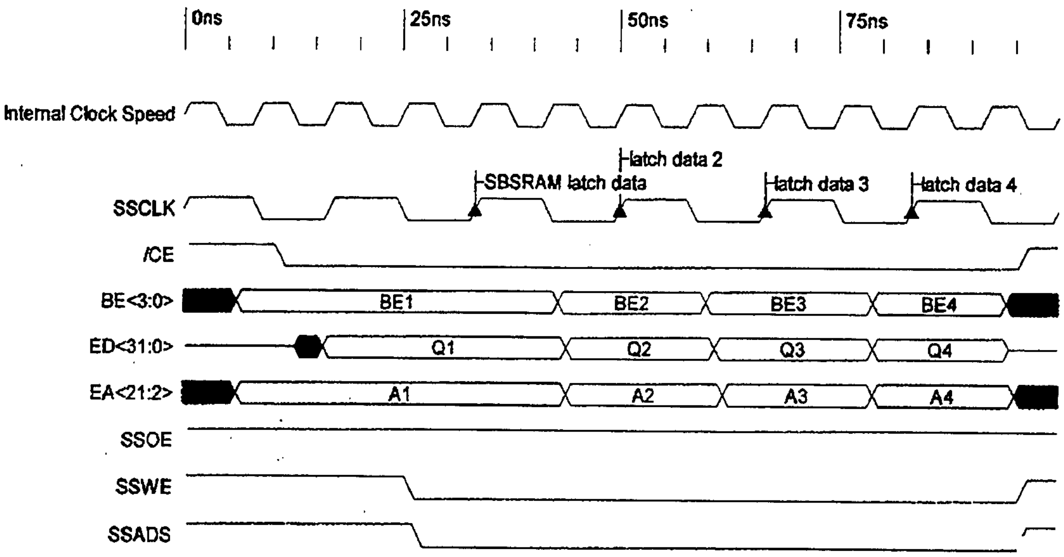

Figure 17. Synchronous Burst SRAM Read Cycle with Voting Delay.

The following precaution is needed for this circuit to ensure false errors are not generated. Placing a pass-gate buffer at the end of the 27-bit input OR gate, that generates the ***EA_ERROR,*** will ensure false positives are not generated. The pass-gate also ensures a sufficient amount of time has passed to allow for voter comparison. The control signal (***ADDVOTER_CONT)*** for the pass-gate is generated by an OR gate using the strobe signals, ***V_ARE, V_AWE, V_SSOE,*** and ***V_SSWE***. One of these four signals is always present during a read or write cycle. The ***EA_ERROR*** is then connected to a simple non-clock SR latch. The ***EA_ERROR*** is connected to the S (set latch). The output of the SR latch is then connected to each processor's ***EXT_INT7***. The SR latch uses the voted ***IACK*** for the reset signal (R input). The simple latch was placed into the circuit to ensure ***EA_ERROR,*** which is connected to the ***EXT_INT7***, is held for the minimum of two internal-clock pulses for each processor. Once the interrupt handling routine commences, the external interrupt is ignored.

In summary, the address, BE, and CE busses were easily monitored to ensure no errors were generated by SEU events. The simplicity of this circuit was due to the fact that these signals were unidirectional

D. TI™ TMS320C6701 DATA BUS INTERFACE

In this section, we will demonstrate that the data bus (***ED<31:0>***) meets the manufacturer's specifications for the '6701.

The peripheral user's manual for the '6701 digital signal processor [Ref. 10] was used in conjunction with the ASRAM and SBSRAM implementation guides [Refs. 14 and 15]. The memory cycle must be split into read and write portions because the data bus is a bi-directional bus. The bi-directional bus must be maintained to allow data to flow to and from external peripheral devices. To allow for data to travel from the external memory to each processor, a bypass pass-gate buffer was placed in series between the memory and the DSP. This pass-gate buffer is also in parallel with voter circuit. Figure 18 represents this simple idea. Using the pass-gate buffers ensures only one gate is driving the bus and a race condition will not occur.

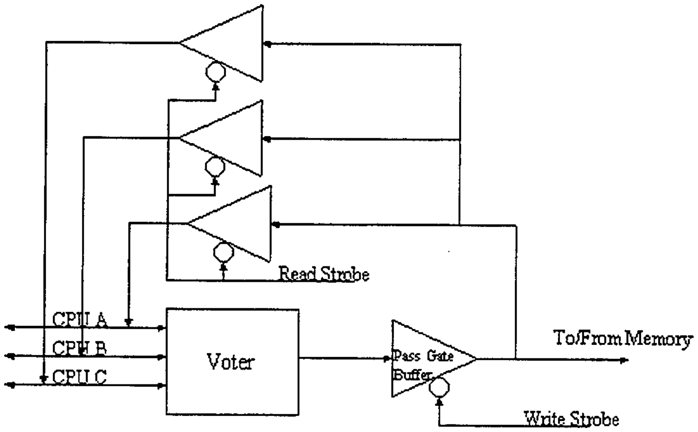

Figure 18. Simple Voter Control and Bypass Circuit.

1. Memory Write Cycle

To generate ***V_ED<31:0>*** a majority voter and error generator was used using the inputs from each processor corresponding signals. Unlike the address generating circuit, the data voter requires both the output of the majority voter and error detector use a pass-gate to control flow. The pass-gate on the output of the error generator is used for the same reason as the pass-gate used on the error generator for the address. The pass-gate on the output of the majority voter is used to control flow on the data bus. Figures 19 and 20 demonstrate the abilities to use asynchronous memory types and the synchronous burst SRAM.

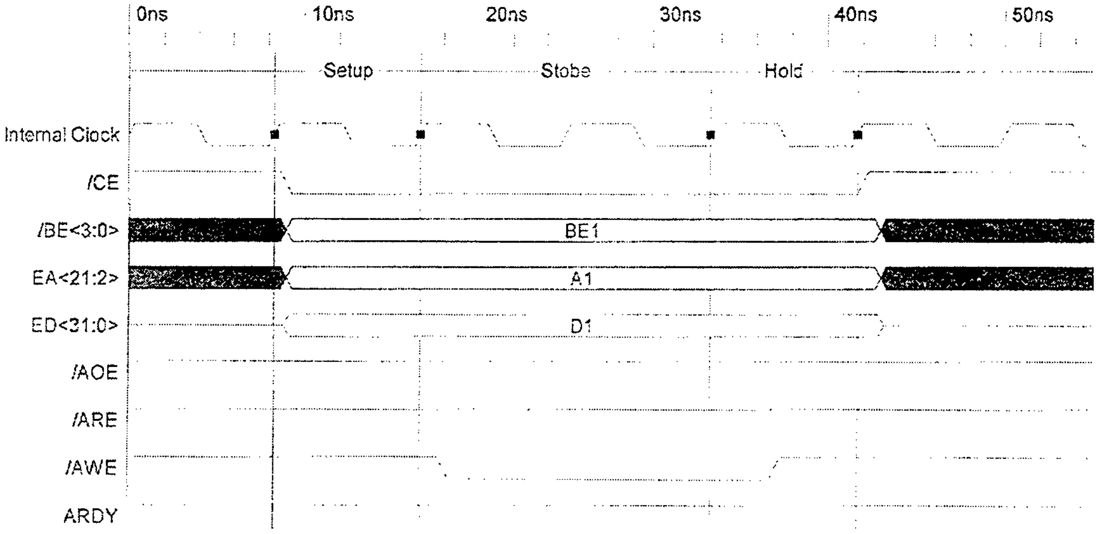

Figure 19. Asynchronous SRAM Write Cycle with Delay of Voting.

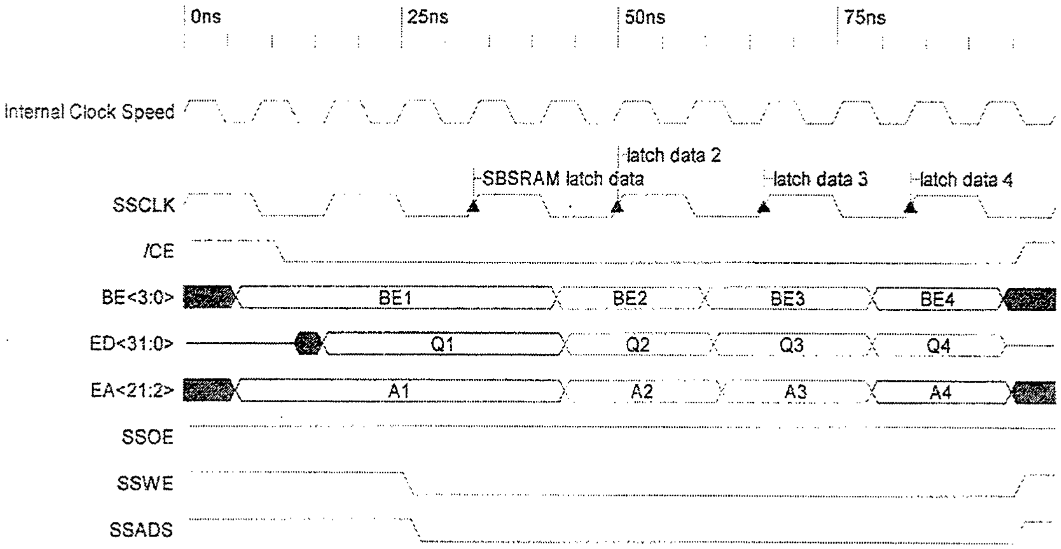

Figure 20. Synchronous Burst SRAM Write Cycle with Delay of Voting.

To generate ***DATA_ERROR*** a 32-input OR gate, that uses the ***ERRORSIG*** from each voter bit circuit, was used. A pass-gate buffer is at the output of the 32 input OR gate. The pass-gate ensures a sufficient amount of time has passed to allow the circuit to settle. The control signal (***WRITE_CONT***) for the pass-gate was generated by an OR gate using the strobe signals ***V_WE,*** and ***V_SSWE***. One of these two signals is always present during a write cycle. ***WRITE_CONT*** was also used to active the pass-gate controlling the flow of data to the external memory locations.

The ***DATA_ERROR*** connected to a clocked D Flip-Flop. The output of the D Flip-Flop is connected to the ***TINP*** for each processor. To clock the flip-flop one of the ***SSCLK*** was used. ***SSCLK*** is used because it runs at half the speed of the CPU, this provides for a long enough pulse to be recognized by the '6701. The flip-flop was placed into the circuit to ensure ***DATA_ERROR,*** which is connected to the ***TINP***, is held for the minimum of two internal-clock pulses for each processor. Each processor will then count the pulses generated by the data error generator. By using the internal counting register, this allows the design to be flexible. Since the assumption is made that data errors were less important than address errors, it is desired to provide the ability to monitor the number data errors. The timer can be easily modified, by modifying the software. This will enable follow-on designers to determine the number of errors that can be accepted on the data bus before a system interrupt must be executed.

2. Memory Read Cycle

The read cycle only requires a means to bypass the voting circuit. This is accomplished by placing pass-gate buffers between the shared memory and the DSP. The control signal (***READ_CONT***) for the pass-gate was generated by an OR gate using the strobe signals, ***V_ARE,*** and ***V_SSOE***. One of these two signals is always present during a read cycle. ***READ_CONT*** was also used to active the pass-gate controlling the flow of data from the shared memory.

3. Addition Considerations for the Read/Write Cycle for SBSRAM

Due to the timing requirement of the SBSRAM cycle, it was determined that the SBSRAM must be operated at half of the processors internal clock rate, in other word at 60 MHz. This was necessary to allow the signals to propagate through the voting circuit. As seen from Figures 17 and 20, address and data are placed on the busses during the falling edge of the output ***SSCLK*** and clock into the memory chip during the rising edge of the ***SSCLK***. This provides for enough time for propagation delay due to the voting circuit and setup time required by the SBRAM memory type. This provides for a safety margin of 6 ns for a write cycle. It was not possible to meet the minimum set time and to allow for propagation delay when the SSCLK was operated at 120 MHz (full internal clock speed). Read cycles are much more tolerant since there is no control buffer impeding the flow of addresses to the memory unit. In addition, only one of the ***SSCLKs*** will be used to clock the SBSRAM.

4. First-in-First-Out (FIFO)

The FIFO is providing a simple means of connecting the DSP circuit to an A/D converter. It is intend to send an interrupt signal to the processors when the FIFOs are half-full. It is located in ***CE1*** memory space. It occupies the entire ***CE1*** memory space, but this can be easily modified if one wants to add a few address lines to the FIFO read enable circuit.

E. TI™ TMS320C6701 HOST PROCESSOR INTERFACE (HPI)

The HPI is used in accordance with the TMS320C6701 user guide [Ref. 10] and the PCI2040 (PCI-HPI bridge) interface implementation guide [Ref. 16]. None of the signals for the HPI bus are voted. The HPI interface bus will be used to determine which processor was at "fault" once an interrupt is generated. The interrupt handling routine will save all general-purpose registers and control registers to the on chip SRAM memory for each processor.

The HPI bus will also be used to load instructions into the memory space of the processors. By using HPI bus to load operating instruction, the host processor will eliminates the need for "boot up" read-only-memory (ROM). This will be useful to test operating systems to control the DSP.

F. IMPACT OF THE TMR SYSTEM

The TMR system has the following impact upon the use of the '6701. It will slow all memory access time by the time required to complete the majority voting process. In this case it was assumed to be 7 ns (from previous design projects) [Ref. 3 and 6]. The 7 ns delay forces the system to operate the SBSRAM at one half the clock rate of the processor regardless of how fast the SBSRAM memory may be. The TMR design will use one of four external interrupt inputs. It will also use one of the two timers. It will also require the modification of an operating system to deal with the external interrupts and excessive data errors. A small portion of on-chip and off-chip memory will need to be reserved for writing the control registers to memory. The off-chip memory is needed for the interrupt service routine to majority vote the processors and re-synchronize the processors and the on-chip memory will be need to help determine why/where an SEU occurred

G. SUMMARY

This chapter covered the implementation of a TMR system using the TI™ TMS320C6701. Appendix B provides the wiring schematic and was developed with the Cadence™ design tool. The timing analysis was done by hand calculations to ensure that all manufacture specifications were met. Further refinement of the system can be made once the type of application is determined (i.e. used for radio signal processing or for image processing).

V. CONCLUSIONS AND FOLLOW-ON RESEARCH

A. SUMMARY

The previous chapter provided the preliminary design for a TMR architecture using the TI™ TMS320C6701 digital signal processor chip and outlined some of the concerns associated with this preliminary design. What follows in this chapter is a presentation of conclusions drawn from the project and possible areas for future work.

This thesis centered on finding a suitable COTS digital signal processor for use in a radiation environment as well as the implications of using that digital signal processor in a Triple-Modulated-Redundant design. The '6701 was selected out of two of the industry's top 32-bit floating-point digital signal processors. To ensure the viability of the '6701, a detailed analysis of possible effects of SEU on the '6701 was performed. From the analysis it was determined that most errors would be detectable on either the data bus or the address bus. Errors that were either not detectable or undetermined were located in proprietary reserved locations within control registers. These bits are often ignored and cannot cause errors, but without detailed design information, it is impossible to determine this with 100 percent certainty. As stated in Chapter III, undetermined and not detectable errors were ignored.

From the detailed analysis, it was demonstrated that the '6701 was suitable for use in a TMR system. Every effort has been made to ensure maximum design flexibility by using several possible memory types with the glueless architecture of the '6701. From this, it was determined that any asynchronous SRAM or synchronous burst SRAM is usable with a TMR architecture. It was also determined that synchronous burst DRAM was unacceptable. Furthermore, some special considerations were taken into account, because of the use of on-chip SRAM. Since data errors would be detected, only after numerous calculations were completed, and it would be pointless then to halt the processor (by perform an interrupt handling routine), data errors would be counted vice causing immediate interrupts. Additionally, it was learned that these data errors would most likely occur during the block transfer of data from internal SRAM to an external memory location. On the other hand, address errors would cause an immediate interrupt and be serviced as a number one priority. Thus, address errors in the '6701 would be treated with the same urgency as in past projects [Ref. 2 and 3]. A system schematic, Appendix B, was also designed to incorporate the ‘6701 into a TMR system. It was designed for maximum flexibility and to minimize restrictions placed upon the system by the TMR architecture that normally is not present in a traditional '6701 implementation.

B. CONCLUSIONS

During this project, several digital signal processors were reviewed for a fault tolerant computing system. The TI™ TMS320C6701 was selected for a triple modular redundant system.

It was found that the ‘6701 architecture is suitable for this application because it is possible to insert a majority voting system on the system busses and still maintain a relatively fast read and write cycle. It was also determined that the TMR system impacted the use of the ‘6701 by slowing the read/write cycle. A detailed, hand analysis of the read/write cycle was preformed to ensure proper timing and bus control. The hand analysis provides proof that the system can be operated at a higher clock speed than current radhard devices.

It was found that the TMR system using the '6701 will operate at 120 MHz internal clock rate compare to the radhard RH-21020 which operates at 40 MHz. The '6701 will be able to operate its SBSRAM memory bus at 40 MHz as compared with the RH-21020 that has an operating external memory bus speed of 20 MHz.

Data errors and address errors should be treated separately. Only address errors should be addressed immediately, to ensure the system does not branch to an undefined or out of sequence location. Data errors are counted. When they exceed a threshold, the design causes a processor interrupt and resynchronizes the system.

C. FOLLOW-ON RESEARCH

To complete this project, the TMS320C6701 must be tested in a radiation environment. Total ionization dose, single event latchup and single event burnout must be determined. The ‘6701 must also have an acceptable single-event-upset rate. If the error rate is too high, the system will spend much of its time handling interrupts and less of its time processing data. The number of SEU errors that the TMR design can tolerate will have to be determined.

Once the ‘6701 has been validated for a radiation environment, the voting design should be transferred into a programmable logic device. An operating system must then be modified to provide an interrupt handling routine that deals with voting errors. Since memory errors can perpetuate for long periods, the operating system should also update constants that are stored in on-chip memory to ensure the constants due not become corrupted. By updating the internal memory locations that store constants with a DMA channel periodically, data errors may be kept to a minimum. Once a system is built, testing should be done to determine the level of data errors that can be permitted before it has serious adverse influences on the system's fault tolerant performance.

APPENDIX A. DETAILED ANALYSIS OF SEU ON THE TMS320C6701

This appendix contains a detailed review how SEU might affect the performance of the '6701. Registers will be examined as to how they might be corrupted by an SEU and how that event will be detected and corrected. This detailed review is provided to ensure all possible effects of SEU can be detected.

Address Mode Register

31 26	25 21	20 16
Reserved	BK1	BK2

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
B7 mode	B6 mode	B5 mode	B4 mode	A7 mode	A6 mode	A5 mode	A4 mode

Figure 21. Address Mode Register (Bits 31:26 shaded).

Affects of an error: Error in the shaded region (Figure 21) will cause an undetermined error.

Detection of error: With the data provided by the technical documentation, it is not possible to determine how to detect an error in this region.

31 26	25 21	20 16
Reserved	BK1	BK2

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
B7 mode	B6 mode	B5 mode	B4 mode	A7 mode	A6 mode	A5 mode	A4 mode

Figure 22. Address Mode Register (Bits 25:16 shaded).

Affects of an error: Error in the shaded region (Figure 22) will cause an address generation error (either causing a circular addressing to terminate too early or late).

Detection of error: It can only be done when addressing an external memory location. Internal memory address errors will be undetectable.

31 26	25 21	20 16
Reserved	BK1	BK2

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
B7 mode	B6 mode	B5 mode	B4 mode	A7 mode	A6 mode	A5 mode	A4 mode

Figure 23. Address Mode Register (Bits 15:0 Shaded).

Affects of an error: Error in the shaded region (Figure 23) will cause an address generation error. If any of the shaded regions go to 11b, the error will result in an undetermined error

Detection of error: Only when addressing an external memory location. Internal memory locations errors will be undetectable.

Control Status Register

31 24	23 16
CPU ID	Revision ID

15 10	9	8	7 5	4 2	1	0
PWRD	SAT	EN	PCC	DCC	PGIE	GIE

Figure 24. Control Status Register (Bits 31:16 Shaded).

Affects of an error: Errors in the shaded region (Figure 24) will cause instructions to be misinterpreted by the processor or cause an interrupt for invalid entry. This will lead to either an address error or data error.

Detection of error: Detectable on either the data or address busses.

31 24	23 16
CPU ID	Revision ID

15 10	9	8	7 5	4 2	1	0
PWRD	SAT	EN	PCC	DCC	PGIE	GIE

Figure 25. Control Status Register (Bits 15:10 Shaded).

Affects of an error: Error in the shaded region (Figure 25) is always read as zero.

Detection of error: No error can occur.

31 24	23 16
CPU ID	Revision ID

15 10	9	8	7 5	4 2	1	0
PWRD	SAT	EN	PCC	DCC	PGIE	GIE

Figure 26. Control Status Register (Bit 9 Shaded).

Affects of an error: Errors in the shaded region (Figure 26) cause an undesirable trap: Usually zero unless a computational unit is saturated (overflows).

Detection of error: When a trap occurs, an instruction will be fetched from memory. This access from memory will usually reside in an external memory location.

31 24	23 16
CPU ID	Revision ID

15 10	9	8	7 5	4 2	1	0
PWRD	SAT	EN	PCC	DCC	PGIE	GIE

Figure 27. Control Status Register (Bit 8 Shaded).

Affects of an error: An error in the shaded region (Figure 27) will result in an ENDIAN format change.

Detection of error: When data is written to external memory locations, the error can be detected. Errors will not be detectable when writing to internal memory.

31 24	23 16
CPU ID	Revision ID

15 10	9	8	7 5	4 2	1	0
PWRD	SAT	EN	PCC	DCC	PGIE	GIE

Figure 28. Control Status Register (Bits 7:2 Shaded).

Affects of an error: An error in the shaded region (Figure 28) will cause the behavior of the '6701's program memory/cache (PCC Field) to change. The DCC field does not apply to '6701.

Detection of error: Change in the program cache may result in address misses, or a cache flush. Eventually, the deviation between processors treatment of internal program memory/cache will result in the address misses no matter what state the PCC change too.

31 24	23 16
CPU ID	Revision ID

15 10	9	8	7 5	4 2	1	0
PWRD	SAT	EN	PCC	DCC	PGIE	GIE

Figure 29. Control Status Register (Bits 1:0 Shaded).

Affects of an error: An error in the shaded region (Figure 29) causes a change in how global interrupts executed or disabled the global interrupts. Potential problem if GIE is changed to zero from a one, all maskable interrupts are disabled.

Detection of error: When an interrupt occurs usually an address is generated and therefore can be detected by the address read.

E1 Phase Program Counter

31 0
(PCE1)

Figure 30. Phase Program Counter (Bits 31:0 Shaded).

Affects of an error: An error in the shaded region (Figure 30) will cause the wrong code to be executed.

Detection of error: When code is not in on chip memory/cache, an address request will go out to external memory location. It will also be detected when the processors go out of lock-step operations.

Floating-Point Adder Configuration Register (FADCR)

Fields used by .L2

31 27	26 25	24	23	22	21	20	19	18	17	16
Reserved	RMode	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN 1

Fields used by .L1

15 11	10 9	8	7	6	5	4	3	2	1	0
Reserved	RMode	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN 1

Figure 31. Float-Point Adder Configuration Register (Bites 31:27, 15:11 Shaded).

Affects of an error: No error occurs if bit fields from Figure 31 are corrupted.

Detection of error: None.

Fields used by .L2

31 27	26 25	24	23	22	21	20	19	18	17	16
Reserved	RMode	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN 1

Fields used by .L1

15 11	10 9	8	7	6	5	4	3	2	1	0
Reserved	RMode	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN 1

Figure 32. Float-Point Adder Configuration Register (Bits 26,25,10,9 Shaded).

Affects of an error: An error in the shaded region (Figure 32) causes rounding errors during computations.

Detection of error: When data is written to external memory locations, the LSB may differ.

Fields used by .L2

31 27	26 25	24	23	22	21	20	19	18	17	16
Reserved	RMode	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN1

Fields used by .L1

15 11	10 9	8	7	6	5	4	3	2	1	0
Reserved	RMode	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN1

Figure 33. Float-Point Adder Configuration Register (Bits 24:16, 8:0 Shaded).

Affects of an error: Errors in the shaded field (Figure 33) will cause either branching error (if any of the values are tested) or a trap may occur.

Detection of error: When a wrong branch is taken or wrong trap taken an address will be requested and detection occurs by an address error.

Floating-Point Auxiliary Configuration Register (FAUCR)

Fields used by .S2

31 27	26	25	24	23	22	21	20	19	18	17	16
Reserved	DIVO	UNORD	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN1

Fields used by .S1

15 11	10	9	8	7	6	5	4	3	2	1	0
Reserved	DIVO	UNORD	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN1

Figure 34. Floating-Point Auxiliary Configuration Register (Bits 31:27,15:11 Shaded).

Affects of an error : No error. (Figure 34)

Detection of error: None.

Fields used by .S2

31 27	26	25	24	23	22	21	20	19	18	17	16
Reserved	DIVO	UNORD	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN1

Fields used by .S1

15 11	10	9	8	7	6	5	4	3	2	1	0
Reserved	DIVO	UNORD	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN1

Figure 35. Floating-Point Auxiliary Configuration Register (Bits 26:16,10:0 Shaded).

Affects of an error: Error field (Figure 35) could cause false branches or no error at all.

Detection of error: If a false branch is taken then it will likely execute wrong code and will make an address request off chip.

Fields used by .M2

31 27	26 25	24	23	22	21	20	19	18	17	16
Reserved	RMode	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN1

Fields used by .M1

15 11	10 9	8	7	6	5	4	3	2	1	0
Reserved	RMode	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN1

Figure 36. Floating-Point Multiplier Configuration Register (Bits 31:27, 15:11Shaded).

Affects of an error: No error. (Figure 36)

Detection of error: No error

Fields used by .M2

31 27	26 25	24	23	22	21	20	19	18	17	16
Reserved	RMode	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN1

Fields used by .M1

15 11	10 9	8	7	6	5	4	3	2	1	0
Reserved	RMode	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN1

Figure 37. Floating-Point Multiplier Configuration Register (Bits 26:25, 10:9 Shaded).

Affects of an error: Errors in the shaded (Figure 37) will cause rounding errors during computations.

Detection of error: When data is written to external memory location.

Fields used by .M2

31 27	26 25	24	23	22	21	20	19	18	17	16
Reserved	RMode	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN1

Fields used by .M1

15 11	10 9	8	7	6	5	4	3	2	1	0
Reserved	RMode	Under	INEX	Over	Info	Inval	DEN2	DEN1	NAN2	NAN1

Figure 38. Float-Point Multiplier Configuration Register (Bits 24:16, 8:0 Shaded).

Affects of an error: Error could cause false branches or no error at all. (Figure 38)

Detection of error: A false branch is taken then it will likely execute the wrong code and will make an address request off chip.

General format for control registers for each computation unit

Operations on the .M

31 29	28	27 23	22 18	17 13	12	11	6	5	4	3	2	1	0
creg	z	dst	src2	src1/cst	x	op	0	0	0	0	0	s	p

Operations on the .S

31 29	28	27 23	22 18	17 13	12	11 6	5	4	3	2	1	0
creg	z	dst	src2	src1/cst	x	op	1	0	0	0	s	p

Operations on the .D

31 29	28	27 23	22 18	17 13	12	6	5	4	3	2	1	0
creg	z	dst	src2	src1/cst	op	1	0	0	0	0	s	p

Operations on the .L

31 29	28	27 23	22 18	17 13	12	11 5	4	3	2	1	0
creg	z	dst	src2	src1/cst	x	op	1	1	0	s	p

Figure 39. General Control Registers (Bits 31:29 Shaded).

Affects of an error: Effects are undetermined. (Figure 39) (Not defined in the manual)

Detection of error: Unknown.

Operations on the .M

31 29	28	27 23	22 18	17 13	12	11 7	6	5	4	3	2	1	0
creg	z	dst	src2	src1/cst	x	op	0	0	0	0	0	s	p

Operations on the .S

31 29	28	27 23	22 18	17 13	12	11 6	5	4	3	2	1	0
creg	z	dst	src2	src1/cst	x	op	1	0	0	0	s	p

Operations on the .D

31 29	28	27 23	22 18	17 13	12 7	6	5	4	3	2	1	0
creg	z	dst	src2	src1/cst	op	1	0	0	0	0	s	p

Operations on the .L

31 29	28	27 23	22 18	17 13	12	11 5	4	3	2	1	0
creg	z	dst	src2	src1/cst	x	op	1	1	0	s	p

Figure 40. General Control Registers (Bits 28:1X Shaded).

Affects of an error: Error will cause either an address calculation error or a data error, because a wrong register is either used to stored data or the data is written into the wrong register. (Figure 40)

Detection of error: If a wrong address is generated, it will cause an address error interrupt when the address is fetched. For data errors, error will not be detected until it is written to external memory location.

Operations on the .M

31 29	28	27 23	22 18	17 13	12	11 7	6	5	4	3	2	1	0
creg	z	dst	src2	src1/cst	x	op	0	0	0	0	0	s	p

Operations on the .S

31 29	28	27 23	22 18	17 13	12	11 6	5	4	3	2	1	0
creg	z	dst	src2	src1/cst	x	op	1	0	0	0	s	p

Operations on the .D

31 29	28	27 23	22 18	17 13	12 7	6	5	4	3	2	1	0
creg	z	dst	src2	src1/cst	op	1	0	0	0	0	s	p

Operations on the .L

31 29	28	27 23	22 18	17 13	12	11 5	4	3	2	1	0
creg	z	dst	src2	src1/cst	x	op	1	1	0	s	p

Figure 41. General Control Registers (Bits X:X,1 Shaded).

Affects of an error: Errors that cause wrong operation will be executed and may cause the processors to go out of synchronization. (Figure 41)

Detection of error: Either address error or data error, and since execution of various opcodes take different amounts clock cycle, the processors will quick go out of sync.

Operations on the .M

31 29	28	27 23	22 18	17 13	12	11 7	6	5	4	3	2	1	0
creg	z	dst	src2	src1/cst	x	op	0	0	0	0	0	s	p

Operations on the .S

31 29	28	27 23	22 18	17 13	12	11 6	5	4	3	2	1	0
creg	z	dst	src2	src1/cst	x	op	1	0	0	0	s	p

Operations on the .D

31 29	28	27 23	22 18	17 13	12 7	6	5	4	3	2	1	0
creg	z	dst	src2	src1/cst	op	1	0	0	0	0	s	p

Operations on the .L

31 29	28	27 23	22 18	17 13	12	11 5	4	3	2	1	0
creg	z	dst	src2	src1/cst	x	op	1	1	0	s	p

Figure 42. General Control Registers (Bit 0 Shaded).

Affects of an error: Error causes either code to be executed in parallel or not in parallel. Has no effect on data or addresses. (Figure 42)

Detection of error: Processors will go out of sync and therefore have address and data errors on the external bus accesses.

Interrupt Service Table Pointer (ISTP)

31 10	9 5	4	3	2	1	0
ISTB	HPEINT	0	0	0	0	0

Figure 43. Interrupt Service Table Pointer (Bits 31:0 Shaded).

Affects of an error: Error causes an interrupt to occur and is going to branch to the interrupt table in memory, it will branch to the wrong location in memory. (Figure 43)

Detection of error: Processor will put an invalid address on the bus conflicting with the other processors.

Interrupt Enable Register (IER)

31 16
Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE15	IE14	IE13	IE12	IE11	IE10	IE9	IE8	IE7	IE6	IE5	IE4	Rsvd	Rsvd	NMEI	1

Figure 44. Interrupt Enable Register (Bits 31:16,3,2 Shaded).

Affects of an error: Unknown. (Figure 44)

Detection of error: Undetermined.

31 16
Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IE15	IE14	IE13	IE12	IE11	IE10	IE9	IE8	IE7	IE6	IE5	IE4	Rsvd	Rsvd	NMEI	1

Figure 45. Interrupt Enable Register (Bits 15:4,1 Shaded).

Affects of an error: Error causes an interrupt (Figure 45)

Detection of error: An interrupt will not occur when the other two processors execute an interrupt and put out an address request to execute that interrupt.

Interrupt Set Register

31 16
Reserved

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IS15	IS14	IS13	IS12	IS11	IS10	IS9	IS8	IS7	IS6	IS5	IS4	Rsvd	Rsvd	Rsvd	Rsvd

Figure 46. Interrupt Set Register (Bits 31:16,3:0 Shaded).

Affects of an error: Unknown. (Figure 46)

Detection of error: Not defined.

31	16
Reserved	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IS15	IS14	IS13	IS12	IS11	IS10	IS9	IS8	IS7	IS6	IS5	IS4	Rsvd	Rsvd	Rsvd	Rsvd

Figure 47. Interrupt Set Register (Bits 15:4 Shaded).

Affects of an error: No error (Figure 47) (Only indicates if an interrupt has occurred.)

Detection of error: No need. (Only used to set interrupt flag register.)

Interrup Clear Register (ICR)

31	16
Reserved	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC15	IC14	IC13	IC12	IC11	IC10	IC9	IC8	IC7	IC6	IC5	IC4	Rsvd	Rsvd	Rsvd	Rsvd

Figure 48. Interrupt Clear Register (Bits 31:16, 3:0 Shaded).

Affects of an error: Unknown. (Figure 48)

Detection of error: Undetermined.

31	16
Reserved	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC15	IC14	IC13	IC12	IC11	IC10	IC9	IC8	IC7	IC6	IC5	IC4	Rsvd	Rsvd	Rsvd	Rsvd

Figure 49. Interrupt Clear Register (Bits 15:4 Shaded).

Affects of an error: None (Only used to clear interrupt flag register.) (Figure 49)

Detection of error: When control register is written to an external memory location.

NMI Return Pointer (NRP)

31	0
NRP	

Figure 50. NMI Return Pointer (Bits 31:0 Shaded).

Affects of an error: Error cause program to branch bake to wrong location. (Figure 50)

Detection of error: Processor out of sync will produce address request that are invalid.

Interrupt Return Pointer (IRP)

31	0
IRP	

Figure 51. Interrupt Return Pointer (Bits 31:0 Shaded).

Affects of an error: Error causes program to branch to a wrong location. (Figure 51)

Detection of error: Processor out of sync will produce address requests that are invalid.

APPENDIX B. THE SCHEMATIC OUT LINE OF THE TMR SYSTEM

This appendix will provide the detailed schematic of the TMR using three TMS320C6701. The highest level of system will be used followed by the subsystem that makes up each block.

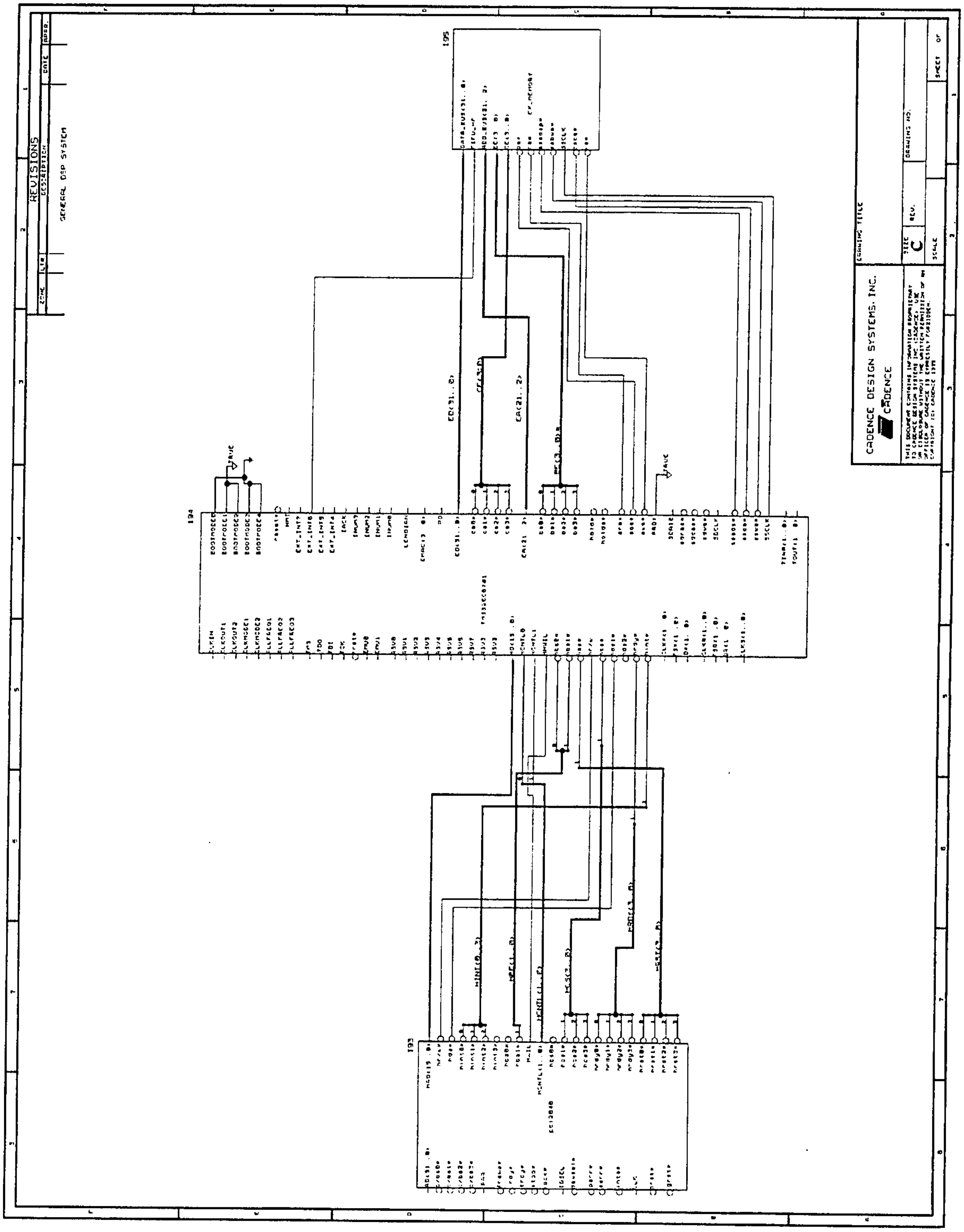

Figure 52. General TMS320C6701 Implementation.

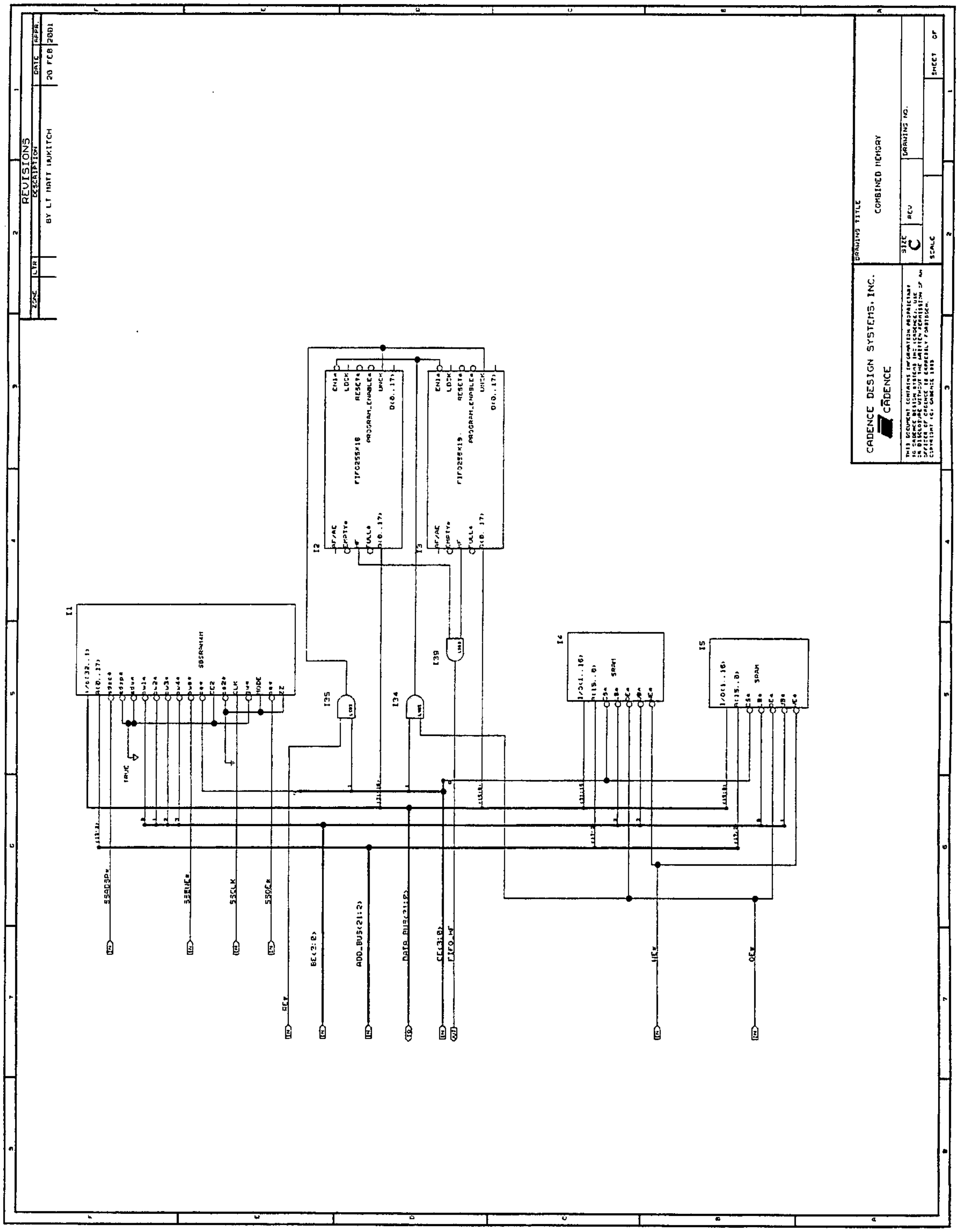

Figure 53. General Memory Architecture.

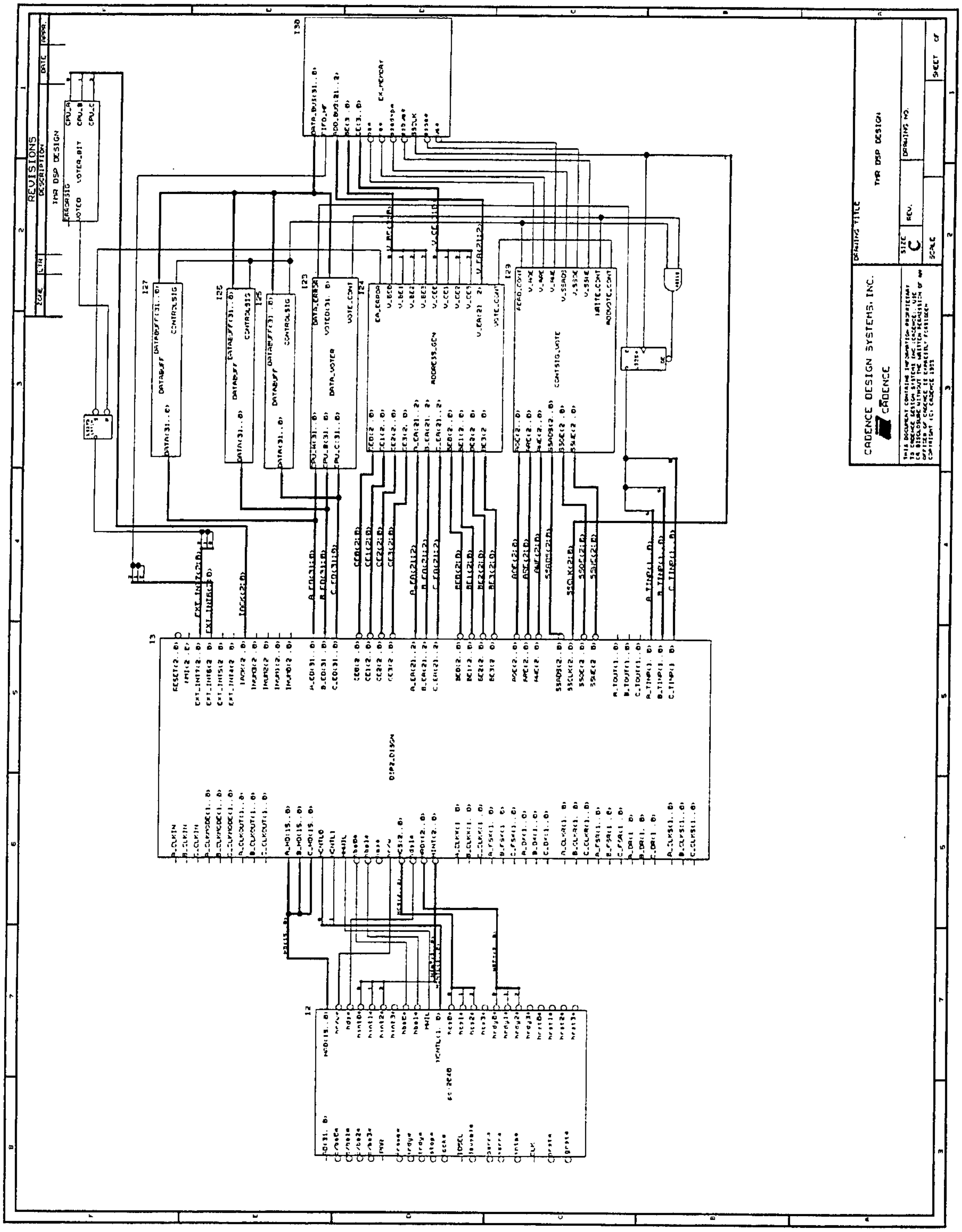

Figure 54. ‘6701 TMR Design.

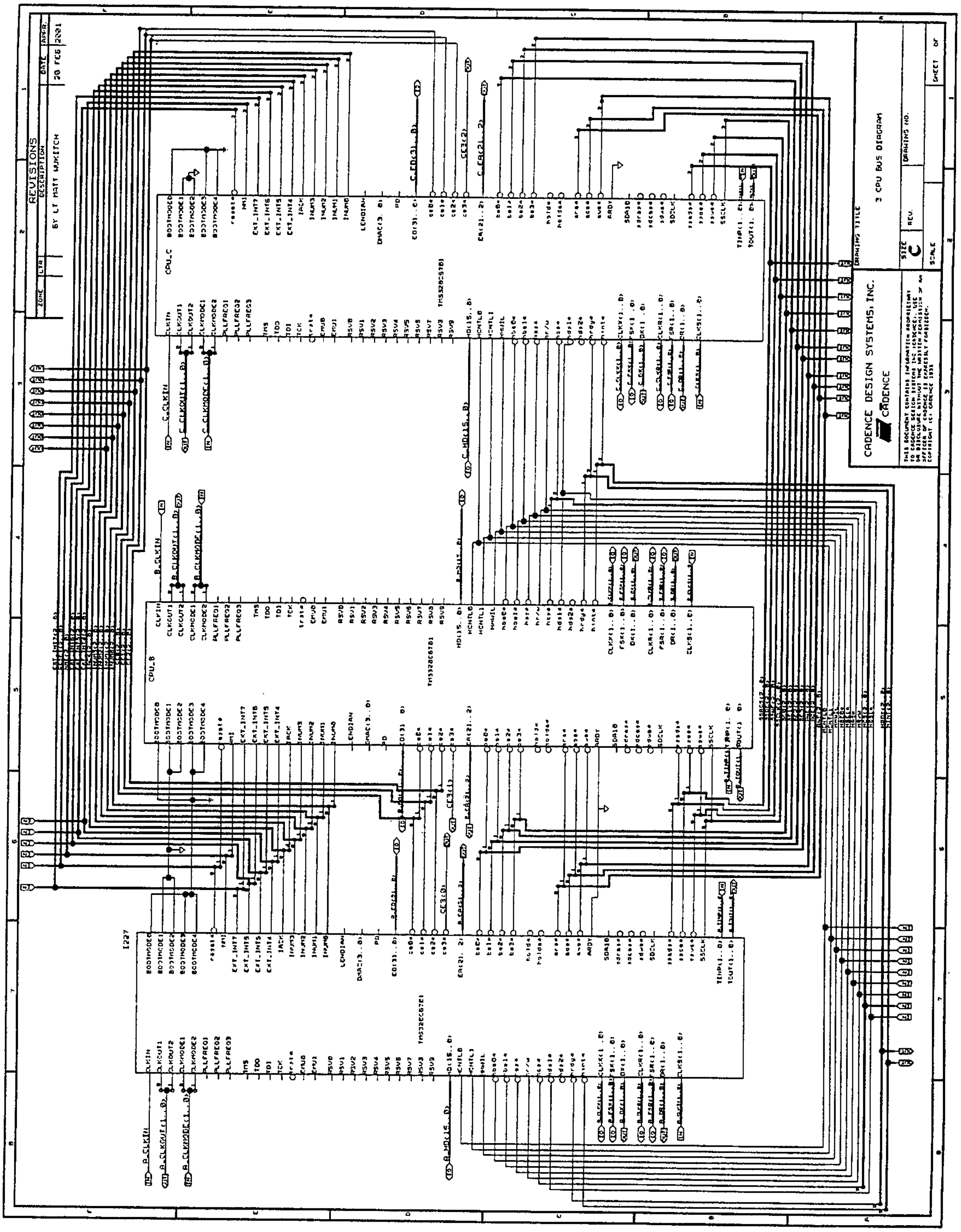

Figure 55. Schematic of Three '6701 Wired in Parallel.

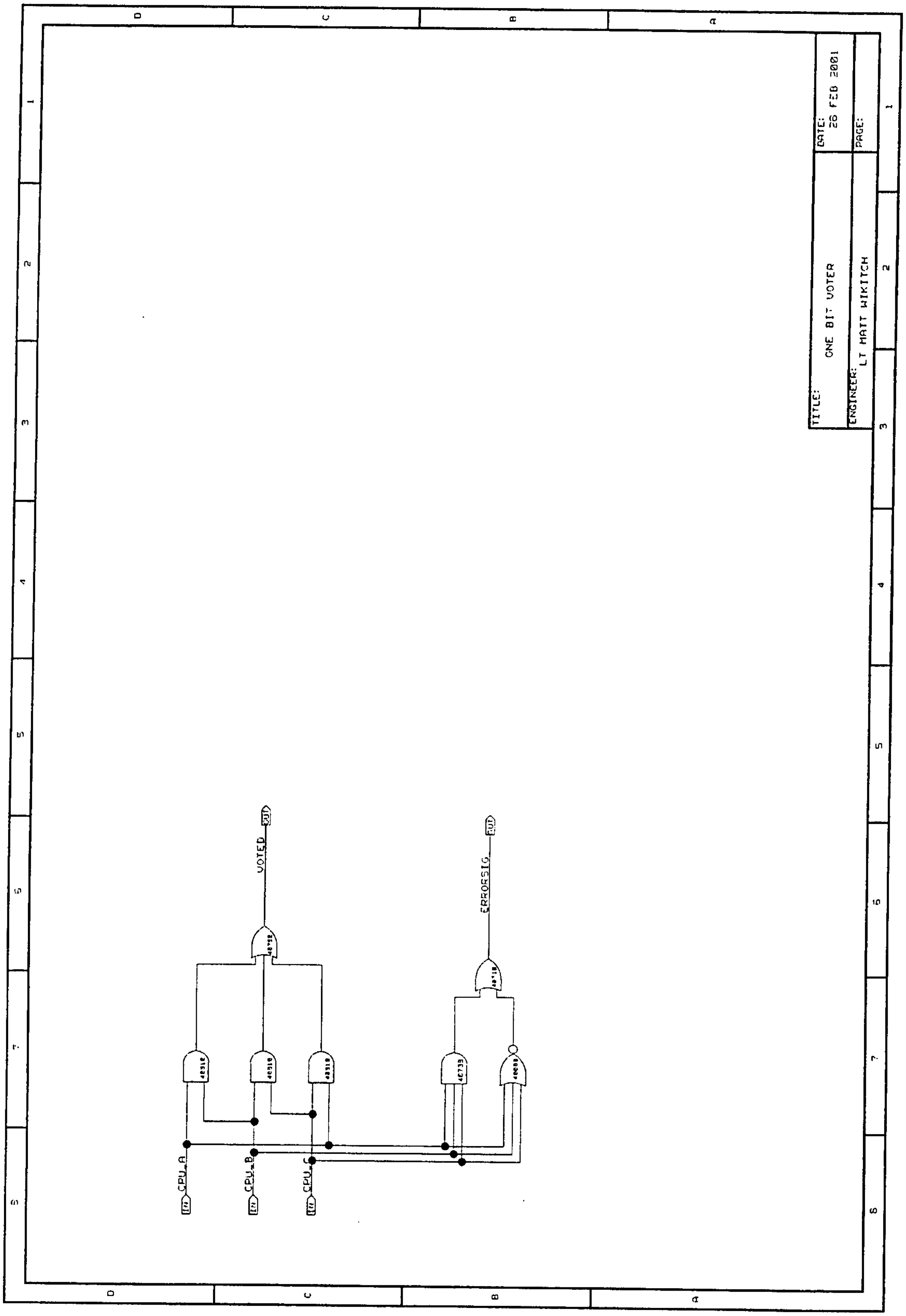

Figure 56. One-Bit Voter.

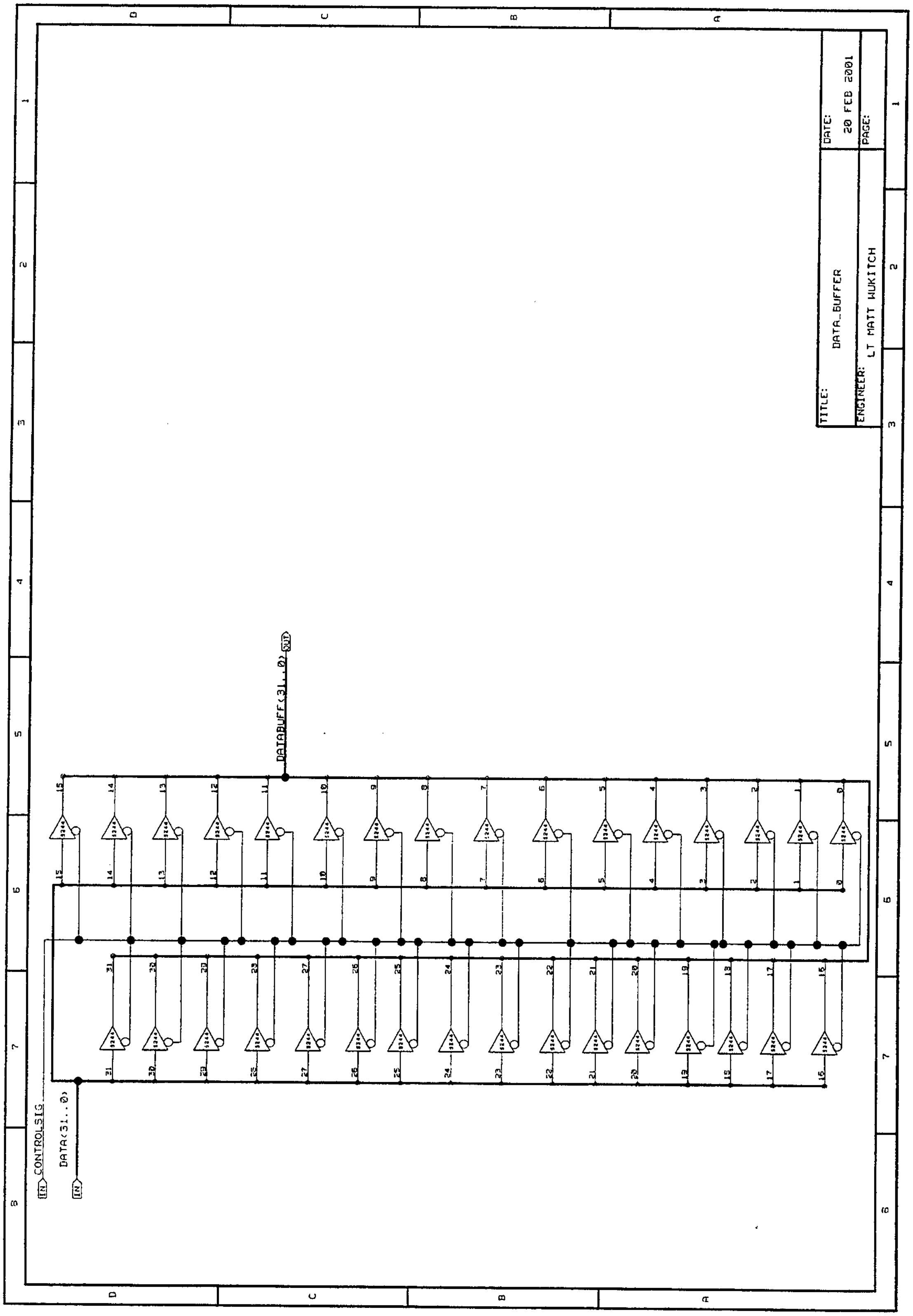

Figure 57. Data Buffer.

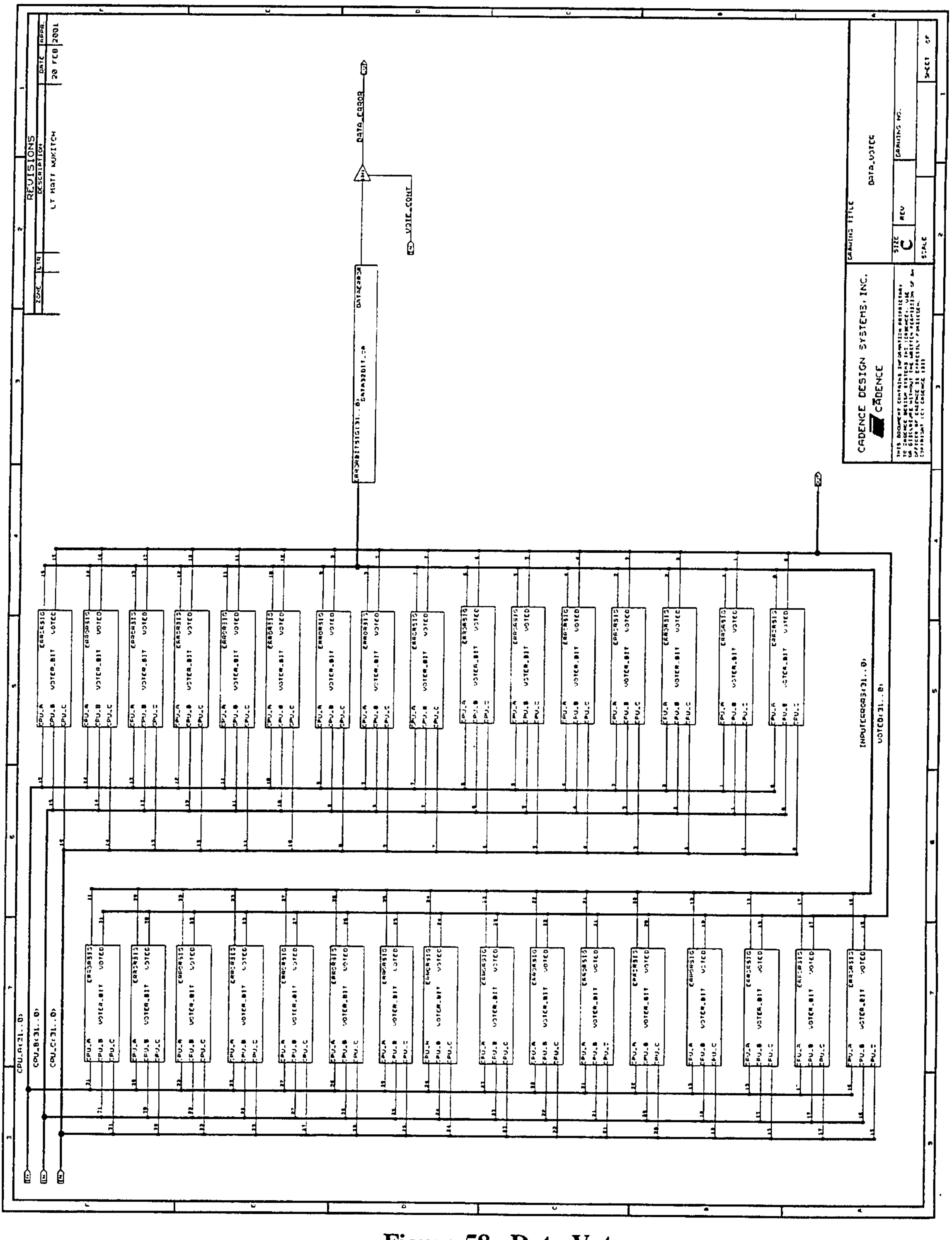

Figure 58. Data Voter.

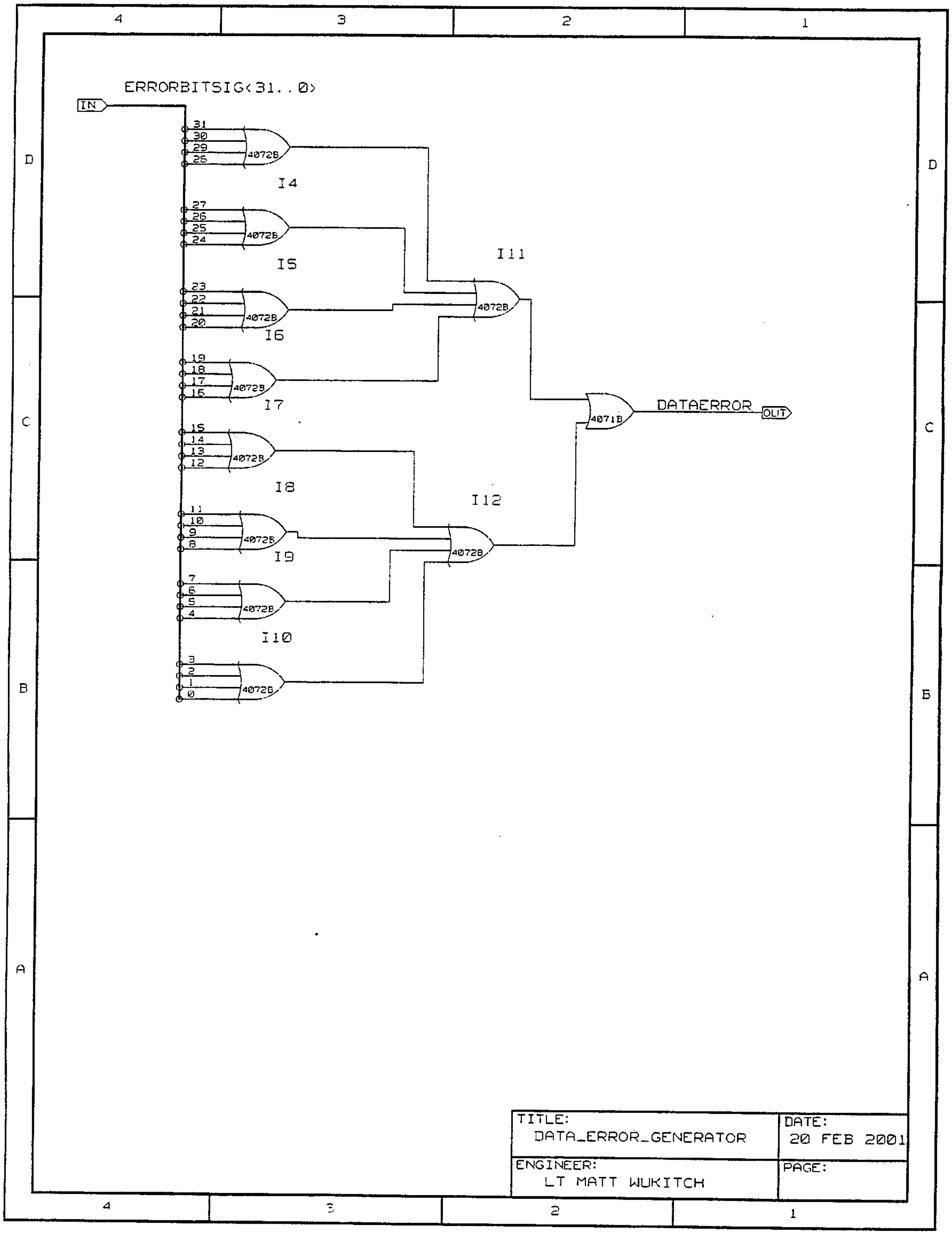

Figure 59. Data Error Generator.

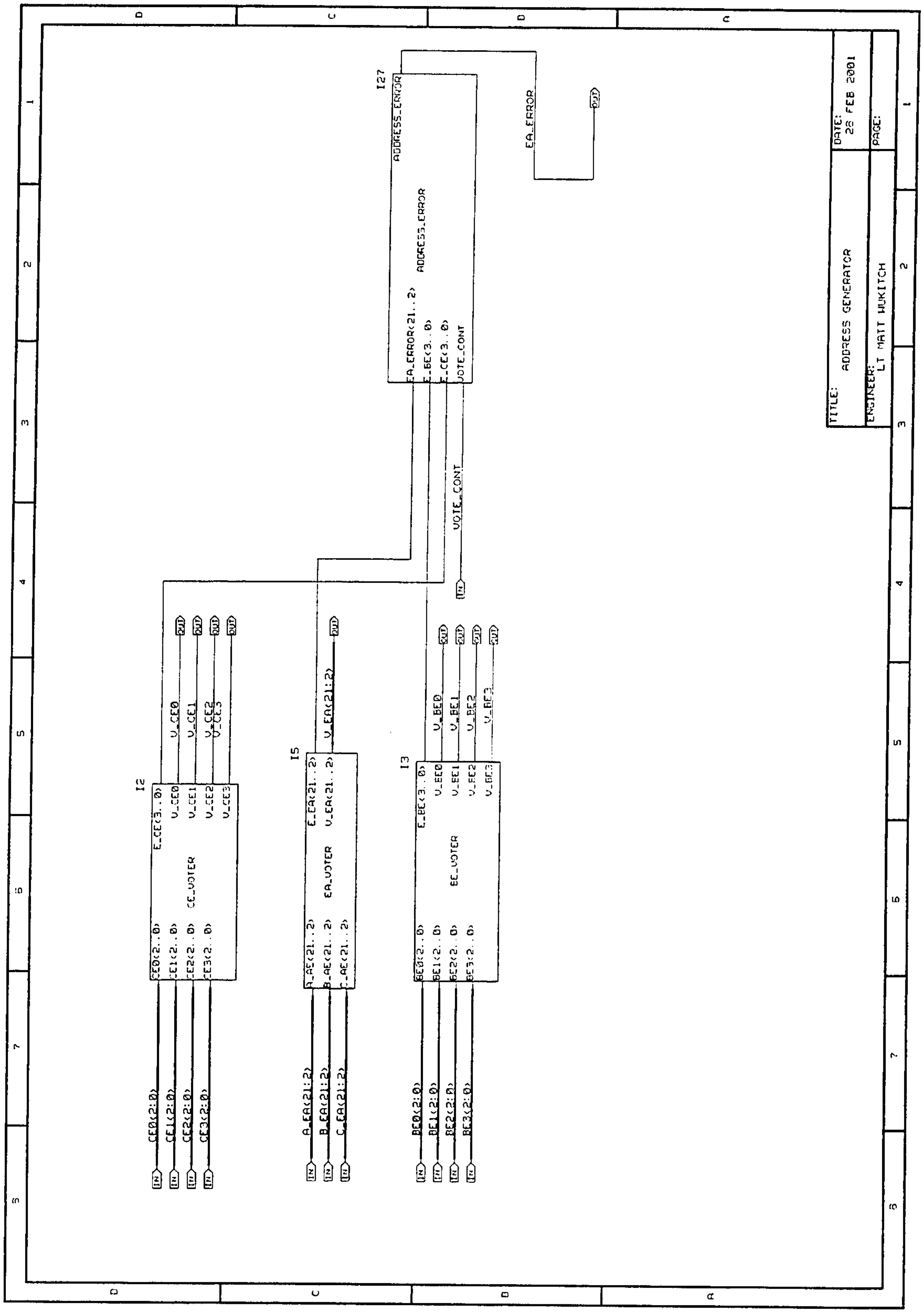

Figure 60. Address Voter.

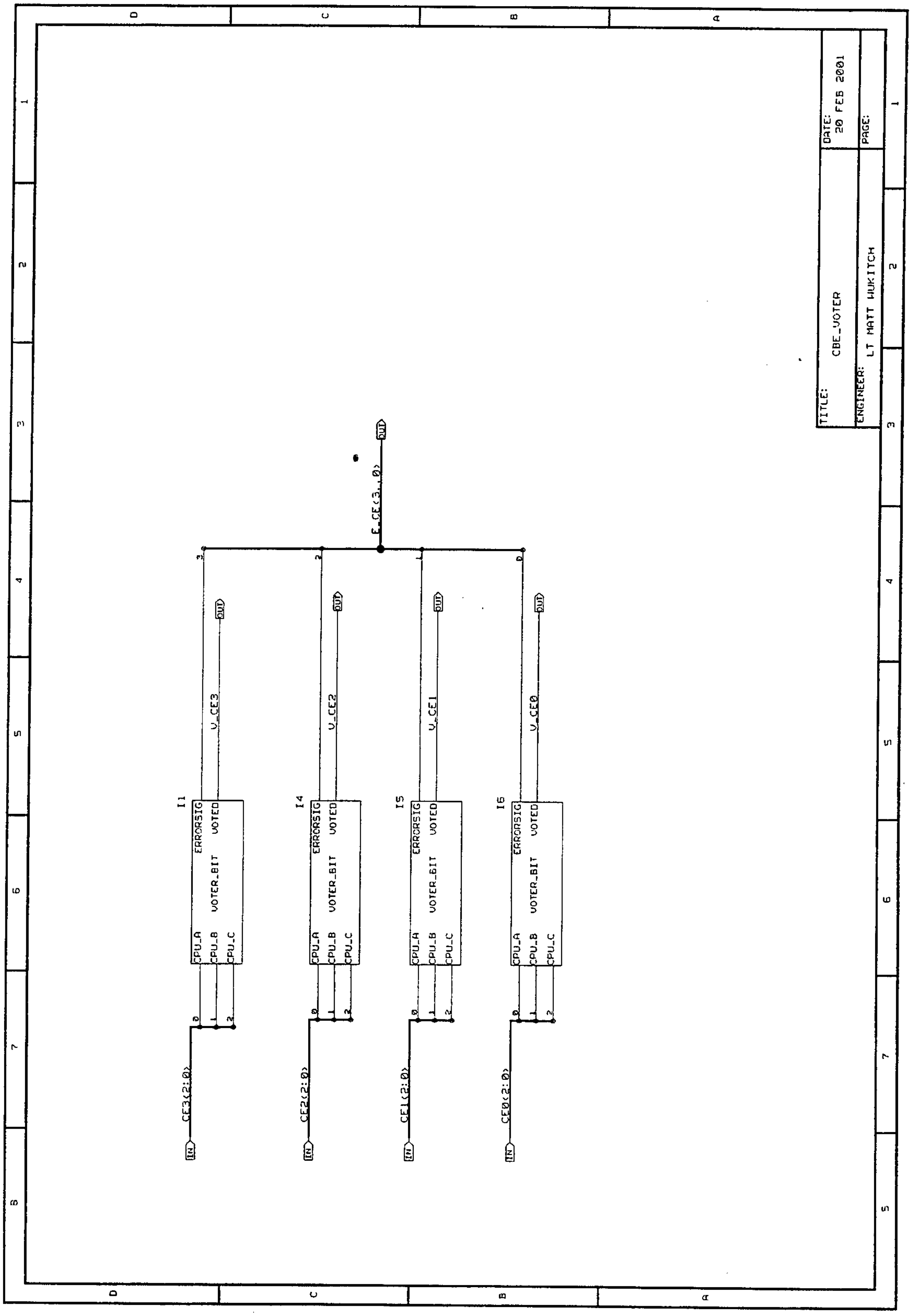

Figure 61. CE Voter.

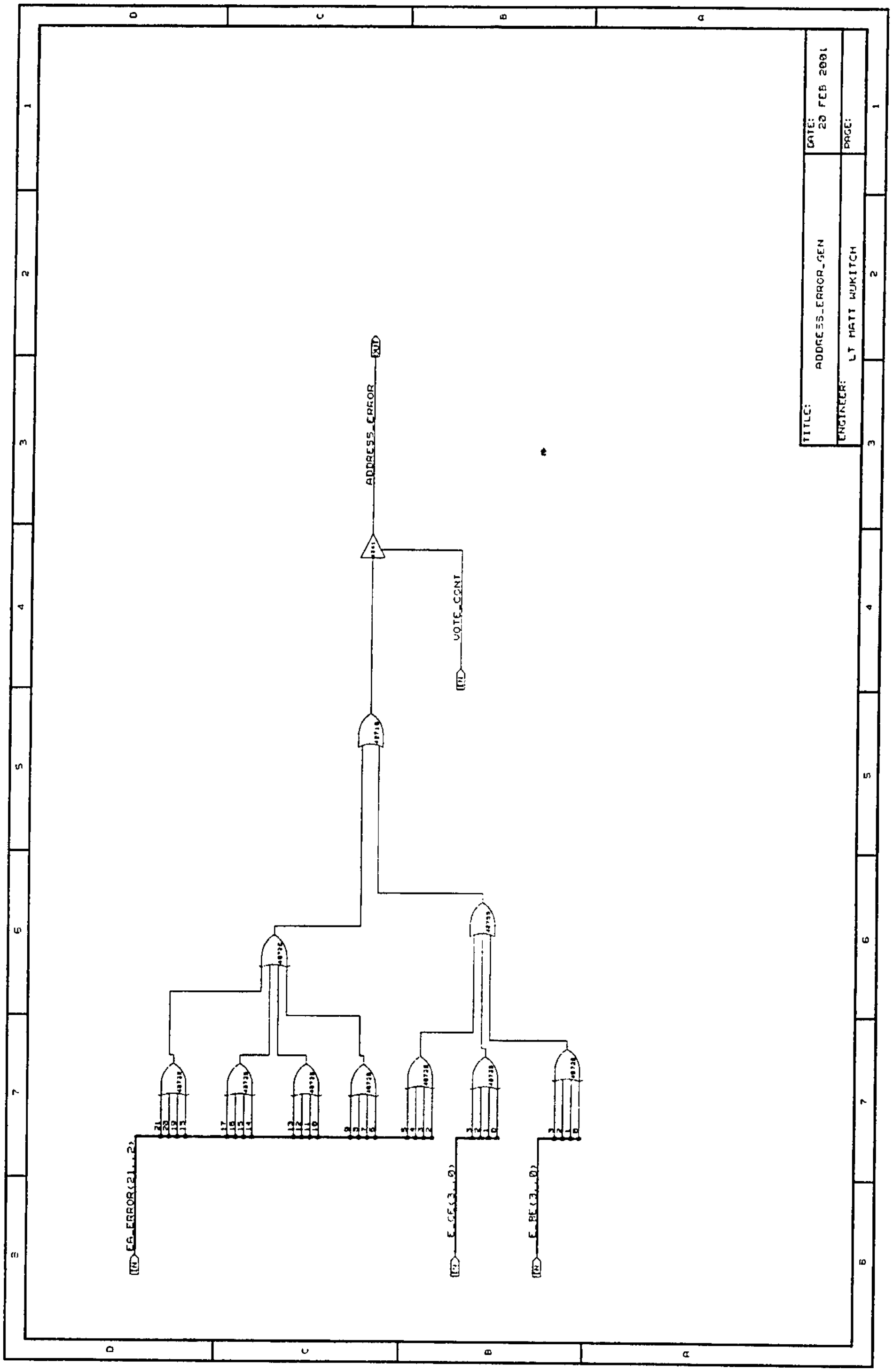

Figure 62. Address Error Generator.

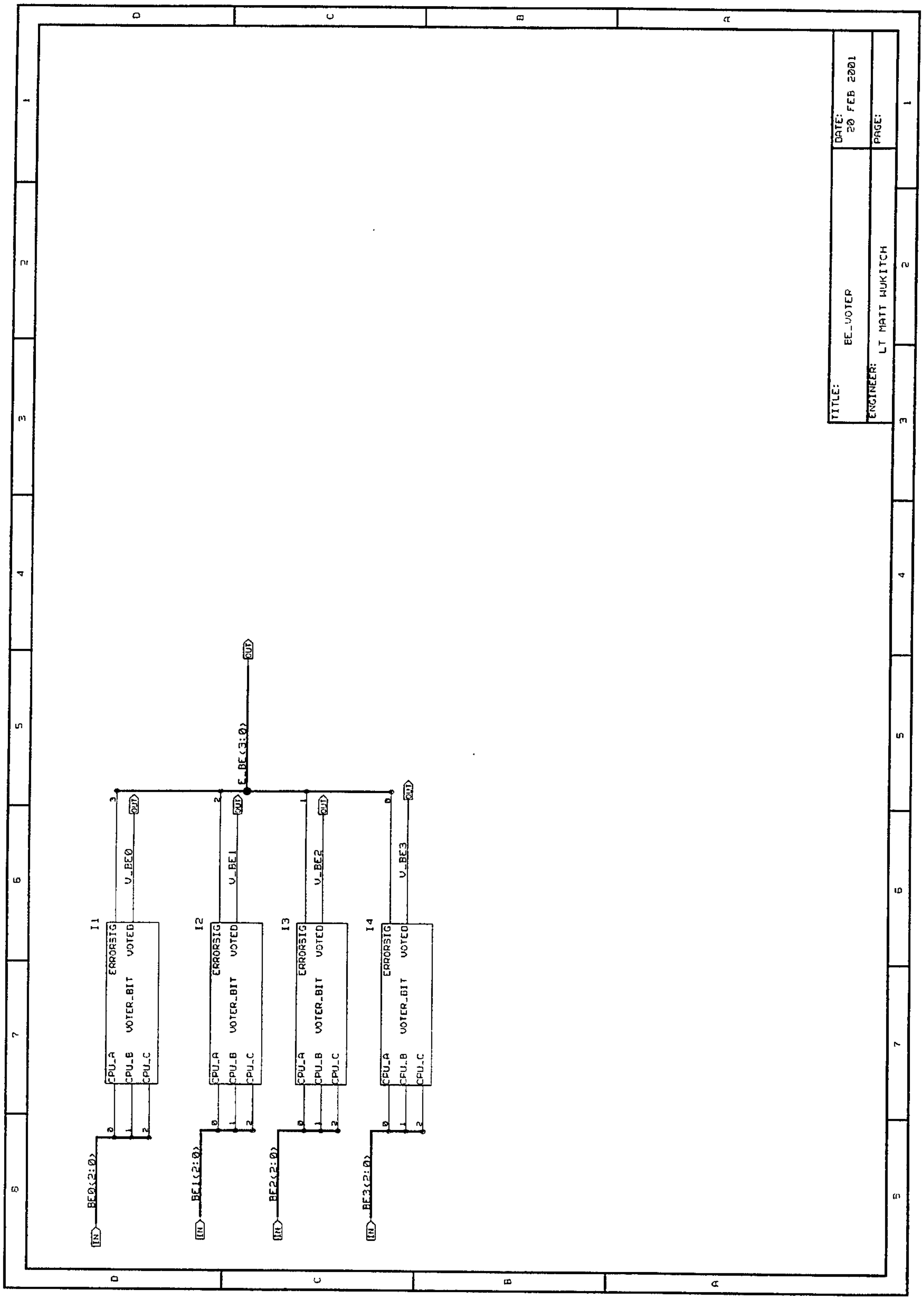

Figure 63. BE Voter.

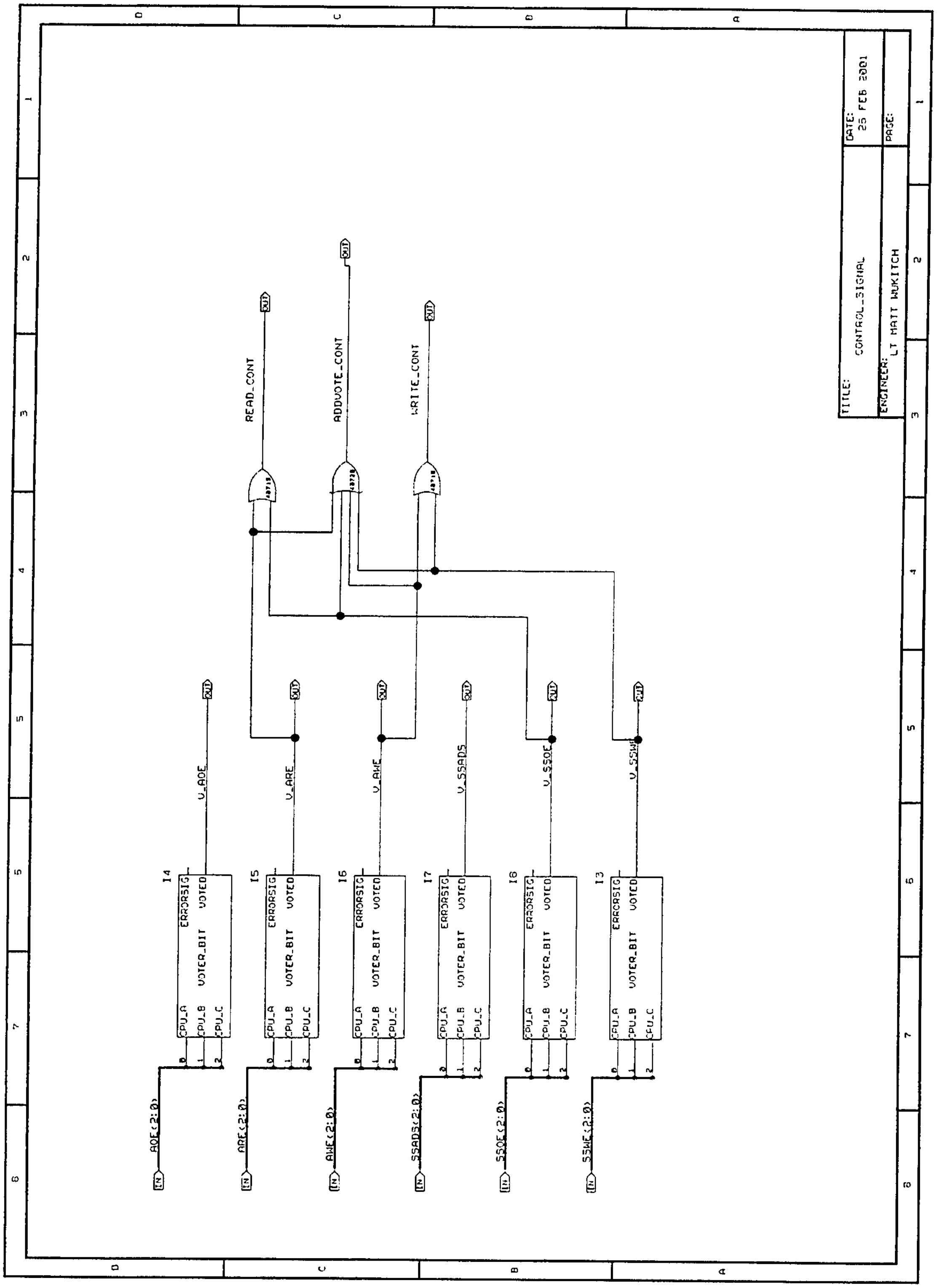

Figure 64. Control Signal Voter.

LIST OF REFERENCES

1. Proakis, J.G. and Manolakis, *Digital Signal Processing*, 3rd Edition, Prentice-Hall, Upper Saddle River, NJ, 1996.

2. Payne, Jr., J. C., *Fault Tolerant Computing Testbed: A Tool for the Analysis of Hardware and Software Fault Handling Techniques*, Master's Thesis, Naval Postgraduate School, Monterey, CA, December 1998.

3. Hofheinz, Damen, *Completion and Testing of ATMR Computing Testbed and Recommendations for a Flight-Ready Follow-on-Design*, Master's Thesis, Naval Postgraduate School, Monterey, CA, December 2000.

4. Summers, David, *Implementation of a Fault Tolerant Computing Testbed: a Tool for the Analysis of Hardware and Software Fault Handling Techniques*, Master's Thesis, Naval Postgraduate School, Monterey, CA, June 2000.

5. Groening, Susan and Whitehouse, Kimberly, *Application of a Fualt-Tolerant Computing for Spacecraft Using Commercial-of-the-shelf Microprocessors*, Master's Thesis, Naval Postgraduate School, Monterey, CA, June 2000.

6. Analog Device Inc. "Analog Devices ADSP-21060 vs. Texas Instruments TMS320C6x," Online, Internet, Available: www.analog.com/publications/whitepapers/products/hh_vs_ti_n/hh_vs_ti_n.html.

7. BAE Systems. "*RH-21020 32/40-Bit IEEE Floating point Radiation Hardened DSP Microprocessor*," Online, Internet, Available: www.baesystems-iews.com/space/pdf/9398.pdf.

8. Sampson, S., Alcorn, C. and Robinson, P., "Radiation Tolerant Digital Signal Transformation," Online, Internet, Available: www.researchdsp.com/papers/21k47.pdf.

9. Whelan, Linda J, ed. *Integrated Circuits for the Space Environment*, Melbourne, FL, Harris Corporation, 1993.

10. Information in an interview with Mr. Kinney, Research Associate at Naval Research Laboratory, October 2000.

11. Texas Instruments, Inc., *TMS320C6701 Floating-Point Digital Signal Processor, Revision 2000*, Literature Number SPRS067E, Dallas, TX, 2000.

12. Texas Instruments, Inc., *TMS320C6000 Peripherals Reference Guide*, Literature Number SPU190C, Dallas, TX, 1999.

13. Texas Instruments, Inc., *TMS320C6000 EMIF to External Asynchronous SRAM Interface*, Literature Number SPRA542, Dallas, TX, 1999.

14. Texas Instruments, Inc., *TMS320C6000 EMIF to External SBSRAM Interface*, Literature Number SPRA533, Dallas, TX, 1999.

15. Texas Instruments, Inc., *TMS320C6000 EMIF to External SDRAM/SGRAM Interface*, Literature Number SPRA433A, Dallas, TX, 1999.

16. Texas Instruments, Inc., *PCI2040 Implementation Guide*, Literature Number SCPU004, Dallas, TX, 1999.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
    8725 John J. Kingman Road, Suite 0944
    Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library 2
    Naval Postgraduate School
    411 Dyer Road
    Monterey, CA 93943-5101

3. Chairman, Code EC 1
    Department of Electrical and Computer Engineering
    Naval Postgraduate School
    Monterey, CA 93943-5121

4. Professor Herschel Loomis Code EC/Lm 2
    Department of Electrical and Computer Engineering
    Naval Postgraduate School
    Monterey, CA 93943-5121

5. Professor Alan Ross Code SP/Ra 2
    Space Systems Academic Group
    Naval Postgraduate School
    Monterey, CA 93943-5110

6. Engineering and Technology Curricular Office, Code 34 1
    Naval Postgraduate School
    Monterey, CA 93943-5109

7. Lieutenant Matthew Wukitch, USN 1
    600 Browns Lane
    Pittsburgh, PA 15237